

3/5/1 (Item 1 from file: 351)

DIALOG(R) File 351: DERWENT WPI

(c) 2000 Derwent Info Ltd. All rts. reserv.

012231289 **Image available**

WPI Acc No: 99-037396/199904

XRPX Acc No: N99-028236

Hash value generating method for data encryption or decryption e.g. for electronic mail - generating hash valves, keys and cipher text rapidly, and when message is sent inputting divided data of message and applying injection extension processing to obtain longer data length

Patent Assignee: HITACHI LTD (HITA)

Inventor: KURUMATANI H; TAKARAGI K

Number of Countries: 026 Number of Patents: 002

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Main IPC	Week
EP 886399	A2	19981223	EP 98110182	A	19980604	H04L-009/32	199904 B
JP 10340048	A	19981222	JP 97149423	A	19970606	G09C-001/00	199910

Priority Applications (No Type Date): JP 97149423 A 19970606

Cited Patents: No-SR.Pub

Patent Details:

Patent	Kind	Lan	Pg	Filing Notes	Application	Patent
--------	------	-----	----	--------------	-------------	--------

EP 886399	A2	E	57			
-----------	----	---	----	--	--	--

Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT
LI LT LU LV MC MK NL PT RO SE SI

JP 10340048	A	53				
-------------	---	----	--	--	--	--

Abstract (Basic): EP 886399 A

The hash value generating method for digital signature or data encryption involves dividing target data into at least two blocks. Character-substitution and/or transposition processing is then performed on any one of the blocks. The processed data is then subjected to a multiplication process so that the data product is longer than the original data length.

The product data is then divided into at least two blocks, and character-substitution and/or transposition processing is again performed on any one of the blocks. The processing of the data may involve an injection extension transformation in which an output value is absolutely different if an input value is different (injection) and the length of the output value is longer than the length of the input value (extension).

USE - Data scrambling e.g. in computer network, e.g. for electronic mail system.

ADVANTAGE - Message is converted to hash value which is difficult to inverse convert.

Dwg.1/27

Title Terms: HASH; VALUE; GENERATE; METHOD; DATA; ENCRYPTION; DECRYPTER; ELECTRONIC; MAIL; GENERATE; HASH; VALVE; KEY; CIPHER; TEXT; RAPID; MESSAGE; SEND; INPUT; DIVIDE; DATA; MESSAGE; APPLY; INJECTION; EXTEND; PROCESS; OBTAIN; LONG; DATA; LENGTH

Derwent Class: W01

International Patent Class (Main): G09C-001/00; H04L-009/32

International Patent Class (Additional): H04L-009/06; H04L-009/30

File Segment: EPI

3/5/2 (Item 1 from file: 347)

DIALOG(R) File 347: JAPIO

(c) 2000 JPO & JAPIO. All rts. reserv.

06056948 **Image available**

HASH VALUE GENERATING METHOD, DATA CIPHERING METHOD, DATA DECIPHERING METHOD, HASH VALUE GENERATING DEVICE DATA CIPHERING DEVICE, AND DATA DECIPHERING DEVICE

PUB. NO.: 10-340048 A]
PUBLISHED: December 22, 1998 (19981222)
INVENTOR(s): TAKARAGI KAZUO
KURUMAYA HIROYUKI
APPLICANT(s): HITACHI LTD [000510] (A Japanese Company or Corporation), JP
(Japan)
APPL. NO.: 09-149423 [JP 97149423]
FILED: June 06, 1997 (19970606)
INTL CLASS: [6] G09C-001/00; G09C-001/00; G09C-001/00; H04L-009/06;
H04L-009/30
JAPIO CLASS: 44.9 (COMMUNICATION -- Other); 44.3 (COMMUNICATION --
Telegraphy)
JAPIO KEYWORD: R131 (INFORMATION PROCESSING -- Microcomputers &
Microprocessors); R303

ABSTRACT

PROBLEM TO BE SOLVED: To disorder data efficiently by performing a multiplying process which makes the length of an output value longer than that of an input value in a hash value generating process.

SOLUTION: Expanded data 107 generated by a data expansion part 102 are divided into 64-bit frames (group of blocks) like a 1st section E(sub 1) 108, a 2nd section E(sub 2) 109... and inputted to a single-shot expansion part 105 in order. The single-shot expansion part 105 performs a single-shot expanding process for the 1st section E(sub 1) 108 by using a 256-bit initial value 110 as a parameter while performing conversion and transposition. For the 2nd section E(sub 2) 109, a single-shot expanding process is performed by using a 1st 256-bit intermediate output which is obtained while conversion and transposition are carried out. Consequently, a 2nd 256-bit intermediate output is calculated. This process is repeated until the frame of the final section is inputted and then the 256-bit intermediate output which is calculated finally is used as a hash value Hash 111.

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平10-340048

(43)公開日 平成10年(1998)12月22日

(51)Int.Cl. ⁶	識別記号
G 0 9 C 1/00	6 5 0
	6 2 0
	6 4 0

F I		
G 0 9 C	1/00	6 5 0 Z
		6 2 0 Z
		6 4 0 D
H 0 4 L	9/00	6 1 1 Z
		6 6 3 Z

審査請求 未請求 請求項の数22 O.L (全 53 頁)

(21)出願番号 特願平9-149423

(22)出願日 平成9年(1997)6月6日

(71)出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72)発明者 宝木 和夫

神奈川県川崎市麻生区王禅寺1099番地 株式会社日立製作所システム開発研究所内

(72) 発明者 車谷 博之

神奈川県横浜市戸塚区戸塚町5030番地 株式会社日立製作所ソフトウェア開発本部内

(74)代理人 弁理士 富田 和子

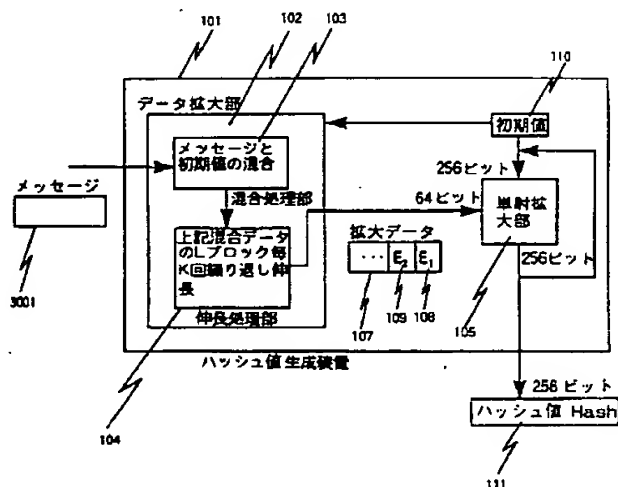
(54) 【発明の名称】 ハッシュ値生成方法、データ暗号化方法、データ復号化方法、ハッシュ値生成装置、データ暗号化装置およびデータ復号化装置

(57) 【要約】

【課題】データ攪乱度の高いハッシュ値や鍵、あるいは暗号文などを、迅速に生成する。

【解決手段】メッセージが与えられたときに、メッセージの分割データを入力し、それよりも長いデータを出力するような単射拡大処理を行う。また、乗算処理、および巡回シフト処理を行う処理を含むハッシュ関数により、ハッシュ値を生成する。

图 1



【特許請求の範囲】

【請求項 1】電子捺印、あるいはデータ暗号化に用いられるハッシュ値生成方法であって、

対象データを少なくとも 2 つのブロックに分割する第一の処理と、

前記第一の処理の結果得られた少なくとも 2 つのブロックのうちのいずれか 1 つに対して、換字および／または転置変換を行う第二の処理と、

前記第二の処理の結果得られたデータに対して、結果が当該データのデータ長よりも長くなるような乗算を行う第三の処理と、

前記第三の処理の結果得られたデータを、さらに少なくとも 2 つのブロックに分割する第四の処理と、

前記第四の処理の結果得られた少なくとも 2 つのブロック各々に対して、換字および／または転置変換を行う第五の処理と、

を含んでいることを特徴とするハッシュ値生成方法。

【請求項 2】電子捺印、あるいはデータ暗号化に用いられるハッシュ値生成方法であって、

対象データを少なくとも 2 つに分割する第一の処理と、

前記第一の処理の結果得られた少なくとも 2 つのブロックのうちの、いずれか少なくとも 1 つに対して、入力値が異なれば出力値も必ず異なり（単射）、かつ出力値の長さが入力値の長さよりも長くなる（拡大）変換である単射拡大変換を行う第二の処理と、

を含んでいることを特徴とするハッシュ値生成方法。

【請求項 3】請求項 2 記載のハッシュ値生成方法であって、

単射拡大変換は、前記第一の処理の結果得られた少なくとも 2 つのブロックのうち、当該単射拡大変換の対象となるデータ以外のデータを、パラメータとすることを特徴とするハッシュ値生成方法。

【請求項 4】請求項 2 記載のハッシュ値生成方法であって、

単射拡大変換は、前記第一の処理の結果得られた少なくとも 2 つのブロックとは別個に設定された初期値をパラメータとすることを特徴とするハッシュ値生成方法。

【請求項 5】請求項 2 記載のハッシュ値生成方法であって、

前記第二の処理は、前記第一の処理の結果得られた少なくとも 2 つのブロックのうちの、いずれか少なくとも 2 つに対して、単射拡大変換を行うものであり、

当該いずれか少なくとも 2 つのうちの 1 つに対して行われる単射拡大変換は、前記第一の処理の結果得られた少なくとも 2 つのブロックとは別個に設定された初期値をパラメータとするものであり、

他の 1 つに対して行われる単射拡大変換は、前記初期値をパラメータとする単射拡大変換の結果をパラメータとするものであることを特徴とするハッシュ値生成方法。

【請求項 6】請求項 2 記載のハッシュ値生成方法であって、

で、

前記第二の処理は、前記第一の処理の結果得られた少なくとも 2 つのブロックのうちの、いずれか少なくとも 2 つに対して、単射拡大変換を行うものであり、かつそのうちの 1 つに対しては、単射拡大変換を 2 回行うものであり、

当該そのうちの 1 つに対して 2 回行われる単射拡大変換は、他の 1 つに対して行われる単射拡大変換の結果をパラメータとすることを特徴とするハッシュ値生成方法。

10 【請求項 7】請求項 2 記載のハッシュ値生成方法であって、

前記単射拡大変換は、入力値に対して換字および／あるいは転置変換を行う第三の処理と、

前記第三の処理の結果得られた入力値に対して、結果が当該データのデータ長よりも長くなるような乗算を行う第四の処理と、

を含んでいることを特徴とするハッシュ値生成方法。

【請求項 8】請求項 2 記載のハッシュ値生成方法であって、

20 前記単射拡大変換は、入力値に対して換字および／あるいは転置変換を行う第三の処理と、

前記第三の処理の結果得られた入力値に対して、巡回シフト計算を行う第四の処理と、

を含んでいることを特徴とするハッシュ値生成方法。

【請求項 9】一定長のデータを暗号変換して一定長の暗号化データを出力するデータ暗号化方法であって、対象データを換字および／または転置変換を行う第一の処理と、

前記第一の処理の結果得られたデータに対して、結果が当該データのデータ長よりも長くなるような乗算を行う第二の処理と、

前記第二の処理の結果得られたデータを、少なくとも 2 つのブロックに分割する第三の処理と、

前記第三の処理の結果得られた少なくとも 2 つのブロック各々に対して、換字および／または転置変換を行う第四の処理と、

含んでいることを特徴とするデータ暗号化方法。

【請求項 10】データを暗号変換して暗号化データを出力するデータ暗号化方法であって、

40 対象データの一部を圧縮変換する第一の処理と、前記第一の処理の結果得られたデータに対して、入力値が異なれば出力値も必ず異なる変換である単射変換を行う第二の処理と、前記第二の処理の結果得られたデータを、暗号化データの一部として出力する第三の処理と、を含む暗号化処理を、当該対象データを構成する全ての部分に対して順次行うものであり、

前記第一の処理は、対象データを構成する最初の一部に対しては、鍵を変換して得られるデータをパラメータとして圧縮変換を行い、対象データを構成する二番目以降の一部に対しては、一つ前の暗号化処理において行われ

た第二の処理の結果をパラメータとして圧縮変換を行うものであり、

前記第二の処理は、単射変換の過程において乗算を2回以上行うものであることを特徴とするデータ暗号化方法。

【請求項11】公開鍵を用いて平文を暗号化する公開鍵暗号を用いたデータ暗号化方法であって、

第一の公開鍵を変換することで得られたデータをパラメータとして、平文に対して暗号化変換を行う第一の処理と、

少なくとも1つの第二の公開鍵に応じたデータと、前記第一の公開鍵を変換することで得られたデータとの間において、前記第二の公開鍵に応じたデータが分かれば、前記第一の公開鍵を変換することで得られたデータを、直接あるいは間接に求めることができる関係式を満たすデータ値を生成する第二の処理と、を含み、

送信すべき暗号化データとして、前記第一の処理の結果得られたデータに、前記第二の処理の結果得られたデータ値を付与したことを特徴とするデータ暗号化方法。

【請求項12】請求項11記載のデータ暗号化方法であって、

前記第一の処理は、生成した乱数と第一の公開鍵とを作用させることで得られたデータを、パラメータとして用いるものであることを特徴とするデータ暗号化方法。

【請求項13】請求項11記載のデータ暗号化方法であって、

前記第一の処理は、複数の区分に分割された平文に対して、区分単位で順次暗号化変換を行うものであり、

かつ、最初の区分に対しては、生成した乱数と第一の公開鍵とを作用させることで得られたデータを、パラメータとして、暗号化変換を行い、

二番目以降の区分に対しては、一つ前の区分に対する暗号化データ生成の過程で発生した中間データを、パラメータとして、暗号化変換を行うものであることを特徴とするデータ暗号化方法。

【請求項14】秘密鍵を用いて暗号文を復号化する公開鍵暗号を用いたデータ復号化方法であって、

当該方法は、請求項11記載のデータ暗号化方法によって得られた暗号文を復号化するものであり、

請求項11記載の第二の公開鍵に応じたデータを、当該第二の公開鍵と対になる秘密鍵から求める第三の処理と、

暗号文に付加されたデータ値と、前記第三の処理で求めたデータとを基に、請求項11において第一の公開鍵を変換することで得られたデータを求める第四の処理と、前記第四の処理で求めたデータをパラメータとして、暗号文を復号する第五の処理と、を含んでいることを特徴とするデータ復号化方法。

【請求項15】請求項14記載のデータ復号化方法であって、

当該方法は、請求項13記載のデータ暗号化方法によって得られた暗号文を復号化するものであり、

前記第五の処理は、複数の区分に分割された暗号文に対して、区分単位で順次復号化変換を行うものであり、

かつ、最初の区分に対しては、前記第四の処理で求めたデータをパラメータとして、復号化変換を行い、

二番目以降の区分に対しては、一つ前の区分に対する復号化データ生成の過程で発生した中間データを、パラメータとして、復号化変換を行うものであることを特徴とするデータ復号化方法。

【請求項16】公開鍵暗号を用いたデータ暗号化・復号化方法であって、

送信側において、

楕円曲線上のベースポイントPのk倍点kPのx座標値と、公開鍵Qのk倍点kQのx座標と、平文とを作用させることで、暗号文を生成するとともに、

当該生成した暗号文に前記kPのx座標値を付加して送信し、

受信側において、

受信した暗号文と、当該暗号文に付加されたkPのx座標値に対応する楕円曲線上の2点のうちのいずれか一方の点Rを公開鍵Qと対になる秘密鍵dでd倍した点dRのx座標値と、を作用させることで、平文を生成することを特徴とするデータ暗号化・復号化方法。

【請求項17】元のデータへの逆変換が困難であるハッシュ値を生成するハッシュ値生成装置であって、

対象データを少なくとも2つのブロックに分割する第一の分割手段と、

前記第一の分割手段で分割され少なくとも2つのブロックのうちのいずれか1つに対して、換字および／または転置変換を行う第一の換字・転置変換手段と、

前記第一の換字・転置変換手段で変換されたデータに対して、結果が当該データのデータ長よりも長くなるような乗算を行う乗算手段と、

前記乗算手段で乗算されたデータを、さらに少なくとも2つのブロックに分割する第二の分割手段と、

前記第二の分割手段で分割された少なくとも2つのブロック各々に対して、換字および／または転置変換を行う第二の換字・転置変換手段と、

を含んでいることを特徴とするハッシュ値生成装置。

【請求項18】元のデータへの逆変換が困難であるハッシュ値を生成するハッシュ値生成装置であって、

対象データを少なくとも2つに分割する第一の分割手段と、

前記第一の分割手段で分割された少なくとも2つのブロックのうちの、いずれか少なくとも1つに対して、入力値が異なれば出力値も必ず異なり（単射）、かつ出力値の長さが入力値の長さよりも長くなる（拡大）変換である単射拡大変換を行う単射拡大変換手段と、

を含んでいることを特徴とするハッシュ値生成装置。

【請求項19】一定長のデータを暗号変換して一定長の暗号化データを出力するデータ暗号化装置であって、対象データに対して換字および／または転置変換を行う第一の換字・転置変換手段と、

前記第一の換字・転置変換手段で変換されたデータに対して、結果が当該データのデータ長よりも長くなるような乗算を行う乗算手段と、

前記乗算手段で乗算されたデータを、少なくとも2つのブロックに分割する分割手段と、

前記分割手段で分割された少なくとも2つのブロック各々に対して、換字および／または転置変換を行う第二の換字・転置変換手段と、

含んでいることを特徴とするデータ暗号化装置。

【請求項20】データを暗号変換して暗号化データを出力するデータ暗号化装置であって、

対象データの一部を圧縮変換する圧縮手段と、前記圧縮手段で圧縮されたデータに対して、入力値が異なれば出力値も必ず異なる変換である単射変換を行う単射変換手段と、前記単射変換手段で変換されたデータを、暗号化データの一部として出力する出力手段と、を含む暗号化手段と、

当該対象データを構成する全ての部分を、前記暗号化手段に順次入力する手段と、を有し、

前記圧縮手段は、対象データを構成する最初の一部に対しては、鍵を変換して得られるデータをパラメータとして圧縮変換を行い、対象データを構成する二番目以降の一部に対しては、一つ前に入力されたデータに対する前記単射変換手段での結果をパラメータとして圧縮変換を行うものであり、

前記単射変換手段は、単射変換の過程において乗算を2回以上行うものであることを特徴とするデータ暗号化装置。

【請求項21】公開鍵を用いて平文を暗号化する公開鍵暗号を用いたデータ暗号化装置であって、

第一の公開鍵を変換することで得られたデータをパラメータとして、平文に対して暗号化変換を行う暗号化手段と、

少なくとも1つの第二の公開鍵に応じたデータと、前記第一の公開鍵を変換することで得られたデータとの間において、前記第二の公開鍵に応じたデータが分かれば、前記第一の公開鍵を変換することで得られたデータを、直接あるいは間接に求めることができる関係式を満たすデータ値を生成するデータ値生成手段と、を含み、

送信すべき暗号化データとして、前記暗号化手段で暗号化されたデータに、前記データ値生成手段で生成されたデータ値を付与することを特徴とするデータ暗号化装置。

【請求項22】秘密鍵を用いて暗号文を復号化する公開鍵暗号を用いたデータ復号化装置であって、

当該装置は、請求項21記載のデータ暗号化装置によつ

て生成された暗号文を復号化するものであり、

請求項21記載の第二の公開鍵に応じたデータを、当該第二の公開鍵と対になる秘密鍵から求める第一のデータ算出手段と、

暗号文に付加されたデータ値と、前記第三の処理で求めたデータとを基に、請求項11において第一の公開鍵を変換することで得られたデータを求める第二のデータ算出手段と、

前記第二のデータ算出手段で求めたデータをパラメータ

10 として、暗号文を復号化する復号化手段と、

を含んでいることを特徴とするデータ復号化装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、電子捺印やデータ暗号化などのコンピュータネットワークにおけるセキュリティを確保する技術に関し、特に、メッセージを逆変換困難な値であるハッシュ値に変換する方法に関する。

【0002】

【従来の技術】電子メールなどのネットワークを介してやり取りされるデータの暗号化方式として、公開鍵暗号化方式がある。この公開鍵暗号化方式による処理の流れを概説すると、次のとおりである。

【0003】ユーザは、自己宛てのメールを暗号化するための公開鍵を、予めメールの送信者に配布しておく。

【0004】メールの送信者は、メールの宛先となるユーザによって予め配布された公開鍵を用いて、メールを暗号化する。そして、暗号化されたメールを、当該メールの宛先へ送信する。

【0005】自己が配布した公開鍵で暗号化された電子メールを受け取ったユーザは、自己が所有する秘密鍵（公開鍵とは異なる数値を持つ）を用いて、当該電子メールを復号化する。

【0006】ところで、この公開鍵暗号化方式は、データの暗号化のみならず、ネットワークを用いた商取引（電子商取引）において、電子的に契約などの正当性を証明する技術である電子捺印（デジタル署名）技術にも応用されている。

【0007】しかしながら、電子捺印技術において、公開鍵暗号だけを用いて長いメッセージに対する電子捺印を生成しようとする、多くの処理時間を要する。このため、メッセージを、一旦、短いデータに圧縮してから、この圧縮されたデータに対して電子捺印を生成する方法が提案されている。

【0008】ここで、圧縮の方法は、通常のデータ圧縮のように、圧縮されたデータから元のメッセージを復元できるように圧縮する必要はない。しかし、ある種の暗号的特性を持つように圧縮する必要がある。このような圧縮を実現する関数として、ハッシュ関数(hash function)が提案されている。

【0009】商取引書等のメッセージ、たとえば文書A:「太郎商会殿 乗用車(カタログNo. 1443)を104万円で購入します。1996年3月10日 吉浦」が、ハッシュ関数への入力データとなる。入力データの長さはどんなに長くても構わない。

【0010】ハッシュ関数は、この入力データに対して暗号変換に似た処理を施すことで一定長の短いデータになるように圧縮する。たとえば、ハッシュ値:283AC9081E83D5B28977が、ハッシュ関数の出力となる。

【0011】このハッシュ値は、メッセージダイジェスト、あるいはフィンガープリント(指紋)とも呼ばれており、理想的には一つの入力データ(メッセージ)に対して世の中に実質上一つしか存在しない。この「世の中に実質上一つしか存在しない」ことを保証するため、ハッシュ値の長さは少なくとも128ビット程度は必要といわれている。より詳しくいえば、ハッシュ関数は、次のような特性をもつことが必要である。

【0012】一方向性(one-way property)あるハッシュ関数の出力値が与えられたとする。この出力値と同じ出力値をもたらすような別のメッセージを求めることが、計算量的に困難でなければならない。

【0013】たとえば、和夫君の誕生日が2月22日であるとすると、和夫君の誕生日と同じ誕生日を持つ別の人の探し出すためには、確率的に言えば、 $365/2=183$ 人程度の人の誕生日を調べればよい。

【0014】このことは、人をメッセージに置き換え、誕生日をハッシュ値に置き換えても、同様に成り立つ。すなわち、ハッシュ値の長さを160ビットとすれば、ハッシュ値の総数は 2^{160} 通りある。あるメッセージのハッシュ値と同じハッシュ値をもつ別のメッセージを探し出すためには、平均して、 $2^{160}/2=2^{159}$ 個のメッセージに調べる必要がある。これは計算量的に困難である。

【0015】衝突回避性(collision free property)メッセージもハッシュ値も何であるかは問わない。とにかく同じハッシュ値となる二つの異なるメッセージを見つけることが計算量的に困難でなければならない。

【0016】たとえば、とにかく誰でも良いから同じ誕生日を持つ二人を探したい。この場合、確率的に言えば、 $365^{1/2}=24$ 人程度の人の誕生日を調べればよい。

【0017】このことは、人をメッセージに置き換え、誕生日をハッシュ値に置き換えても、同様に成り立つ。すなわち、ハッシュ値の長さを160ビットとすれば、同じハッシュ値をもつ二つの異なるメッセージ(どんなメッセージでもよい)を探し出すためには、平均して、 $2^{160}/2=2^{159}$ 組程度のメッセージを調べる必要がある。この数は、一方向性の場合と比べるとかなり小さい。しかし、依然、計算量的には困難である。

【0018】上記のような特性を必要とするハッシュ関数を実現する方法として、種々の方法が発表されているが、現在、換字と転置の繰り返しを行うことでハッシュ値を得る方法が主流になっている。その処理の仕組みを示す従来例が、次の文献1において開示されている。

【0019】ISO/IEC 10118-2, "Information technology - Security techniques - Hash-functions: Part 2: Hash-functions using an n-bit block cipher algorithm" (1994)

10 上記文献1に開示されているハッシュ関数について、図面を用いて説明する。

【0020】図27は従来のハッシュ関数を説明するための図であり、図27(a)は一般的なハッシュ関数の処理の流れを説明するための図、図27(b)は図27(a)に示す換字・転置繰り返し処理3005に、DES(Data Encryption Standard)のような暗号化関数を用いた場合の処理の流れを説明するための図である。

【0021】図27(a)において、圧縮しようとしているメッセージ3001は、第1区分P13002、第2区分P23003、・・・というように、一定の長さ毎に分割され、順次、ハッシュ関数3007に入力される。

【0022】ハッシュ関数3007は、初期値3004をパラメータとして、第1区分P13002に対して換字・転置繰り返し処理3005を施す。これにより、第1番目の中間出力を算出する。

【0023】次に、その第1番目の中間出力をパラメータ(初期値3004の代わり)にして、第2区分P23003に対して換字・転置繰り返し処理3005を施す。これにより、第2番目の中間出力を算出する。

【0024】上記の処理を最後の区分のデータが入力されるまで繰り返すことで、最後に算出された中間出力を、ハッシュ値Hash3006として用いる。

【0025】ここで、上記文献1では、換字・転置繰り返し処理3005として、米国暗号標準DESのような暗号化関数(ブロック暗号)を用いている。このようなハッシュ関数は、「ブロック暗号を利用したハッシュ関数」と呼ばれおり、ISO(International Organization for Standardization)で標準化済みである。

40 【0026】この「ブロック暗号を利用したハッシュ関数」の詳細は次のとおりである。

【0027】図27(b)において、初期値3004を変換関数3008で変換したものを、パラメータとして、第1区分P13002を暗号化関数3009に入力する。そして、暗号化関数3009での暗号化結果と、第1区分P13002との間で、ビット毎に排他的論理和3010を施す。これにより、換字・転置繰り返し処理3005での第1番目の中間出力を算出する。

50 【0028】次に、第1番目の中間出力をフィードバックして、これを変換関数3008で変換したものをパラ

メータとして、第2区分P₂3003を暗号化関数3009に入力する。そして、暗号化関数3009での暗号化結果と、第2区分P₂3003との間で、ビット毎に排他的論理和3010を施す。これにより、換字・転置繰り返し処理3005での第2番目の中間出力を算出する。

【0029】上記の処理を最後の区分のデータが入力されるまで繰り返すことで、最後に算出された中間出力を、ハッシュ値Hash3006として用いる。

【0030】図27(b)に示す「ブロック暗号を利用したハッシュ関数」において、暗号化関数3009としてDESなどを用いる場合、通常、第1区分P₁3002、第2区分P₂3003、・・・の各区分の長さ、および、換字・転置繰り返し処理3005の出力の長さは、各々64ビットとなる。したがって、ハッシュ値Hash3006の長さも64ビットとなる。

【0031】この「ブロック暗号を利用したハッシュ関数」の特徴は、メッセージの各区分P₁3002、P₂3003、・・・の長さ、換字・転置繰り返し処理3005の出力の長さが等しいことである。

【0032】なお、換字・転置繰り返し処理3005として、DESのような暗号化関数を用いないハッシュ関数も提案されている。このようなハッシュ関数は、「専用ハッシュ関数」と呼ばれおり、インターネット標準のMD5や、ISOで標準化中のSHA-1、RIPEMD-160などがある。

【0033】このうち、MD5については、次の文献2で開示されている。

【0034】R.Rivest, "The MD5 Message-Digest Algorithm," IETF RFC 1321 (1992)

なお、MD5の処理の流れ自体は、図27(a)に示すものと同じであるので、この図を用いて説明する。

【0035】まず、圧縮しようとしているメッセージ3001は、第1区分P₁3002、第2区分P₂3003、・・・というように、512ビット毎に分割され、順次、ハッシュ関数3007に入力される。

【0036】ハッシュ関数3007は、128ビットの初期値3004をパラメータとして、第1区分P₁3002に対して、単純な換字・転置繰り返し処理3005を施す。これにより、128ビットの第1番目の中間出力を算出する。

【0037】次に、その第1番目の中間出力をパラメータ(初期値3004の代わり)にして、第2区分P₂3003に対して、単純な換字・転置繰り返し処理3005を施す。これにより、128ビットの第2番目の中間出力を算出する。

【0038】上記の処理を最後の区分のデータが入力されるまで繰り返すことで、最後に算出された128ビットの中間出力を、ハッシュ値Hash3006として用いる。

【0039】この「専用ハッシュ関数」の特徴は、メッセージの各区分P₁3002、P₂3003、・・・の長さよりも、換字・転置繰り返し処理3005の出力の長さが短いことである。

【0040】

【発明が解決しようとする課題】以上説明した従来技術には、以下のような問題点がある。

【0041】(1) 従来提案されているハッシュ関数の問題点

「ブロック暗号を利用したハッシュ関数」の問題点
上述したように、「ブロック暗号を利用したハッシュ関数」では、DESなどの暗号化関数(ブロック暗号)を用いている。ブロック暗号は、通常、入力および出力のデータ長が64ビットである。このため、ハッシュ値も64ビットとなる。一方、上述したように、一つの入力データ(メッセージ)に対して、ハッシュ値が、「世の中に実質上一つしか存在しない」ことを保証するためには、ハッシュ値の長さを、少なくとも128ビット程度以上にする必要があるといわれている。

【0042】したがって、「ブロック暗号を利用したハッシュ関数」において、128ビットのハッシュ値を得ようとする場合、ブロック暗号への入力データ(64ビット)各々に対して、初期値などを変えて、当該ブロック暗号処理を2回行う必要がある。すなわち、ブロック暗号への入力データ(64ビット)各々に対して、出力(64ビット)を2回算出する必要がある。これでは、ハッシュ値生成の処理速度が遅くなるという不満が生じる。

【0043】「専用ハッシュ関数」の問題点
「専用ハッシュ関数」では、「ブロック暗号を利用したハッシュ関数」と異なり、メッセージを分割したデータ各々に対して、換字・転置繰り返し処理を2回行わなくても、128ビットのハッシュ値を得ることができる。

【0044】しかしながら、「専用ハッシュ関数」では、上述したように、メッセージを分割したデータ各々に対して、単純な換字・転置繰り返し処理を施すことで、ハッシュ値を得ている。ここで、換字・転置繰り返し処理の出力値の長さ(上記の例では、512ビット)は、その入力値の長さ(上記の例では、128ビット)より短い。すなわち、換字・転置繰り返し処理において、圧縮が行われていることになる。

【0045】このため、たとえば、512ビット毎に複数のデータに分割した場合に、最後の区分のデータのみが異なるような2つのメッセージを想定した場合、単純な換字・転置繰り返し処理により最後の区分のデータ(512ビット)を128ビットの出力に圧縮する過程において、その出力(すなわちハッシュ値)が同じになる確立が高まる。これでは、衝突回避性を劣化させてしまう。

【0046】なお、上記で述べた問題点は、ハ

ッシュ関数を電子捺印に用いた場合にのみ生じる問題ではない。たとえば、データの暗号化方式に用いた場合でも、同様に生じる問題である。

【0047】(2) 公開鍵暗号方式の問題点

上述したように、公開鍵暗号を用いて長いデータを暗号化しようとする、多くの処理時間を要する。

【0048】 公開鍵暗号方式を電子メールなどのデータ暗号化に適用した場合、同じ電子メールを複数の宛先に暗号化して送信しようとする、送信者は、各宛先毎に、当該宛先から予め配布されて公開鍵を用いて暗号化処理を行わなければならない。すなわち、電子メールの暗号化処理を宛先の数だけ繰り返し行わなければならない。

【0049】 一方、受信者は、間違ってファイルから復号鍵を消去するなどして、復号鍵を紛失してしまった場合、自己宛てに送られてきた、自己が配布した公開鍵で暗号化された電子メールを復号することができなくなってしまう。

【0050】 本発明は上記事情に鑑みてなされたものであり、本発明の目的はデータ攪乱度の高いハッシュ値や鍵、あるいは暗号文などを、迅速に生成する方法および装置を提供することにある。

【0051】 また、本発明の他の目的は、間違ってファイルから復号鍵を消去するなどして、復号鍵を紛失してしまった場合でも、自己宛てに送られてきた暗号化データを、他の二人以上の受信者が協力することで復号することが可能なデータ暗号化方法および装置を提供することにある。

【0052】

【課題を解決するための手段】 上記課題を解決するために、本発明のハッシュ値生成方法および装置は、対象データを少なくとも2つのブロックに分割する第一の構成要素と、前記第一の構成要素の結果得られた少なくとも2つのブロックのうちのいずれか1つに対して、換字および/または転置変換を行う第二の構成要素と、前記第二の構成要素の結果得られたデータに対して、結果が当該データのデータ長よりも長くなるような乗算を行う第三の構成要素と、前記第三の構成要素の結果得られたデータを、さらに少なくとも2つのブロックに分割する第四の構成要素と、前記第四の構成要素の結果得られた少なくとも2つのブロック各々に対して、換字および/または転置変換を行う第五の構成要素と、を含む。

【0053】 本発明のハッシュ値生成方法および装置では、上述したように、ハッシュ値生成過程において、入力値に対して出力値の長さが長くなるような乗算を行っている。乗算処理によれば、出力値のビット各々が入力値のビット各々の影響を受けるので、データの攪乱を効率よく行うことができる。

【0054】 ところで、乗算処理は、最近のマイクロプロセッサの進展により取り分け処理速度が向上してい

る。したがって、データ攪乱度の高いハッシュ値の生成を迅速に行うことができる。

【0055】 また、本発明の他のハッシュ値生成方法および装置は、対象データを少なくとも2つに分割する第一の構成要素と、前記第一の構成要素の結果得られた少なくとも2つのブロックのうちの、いずれか少なくとも1つに対して、入力値が異なれば出力値も必ず異なり(単射)、かつ出力値の長さが入力値の長さよりも長くなる(拡大)変換である単射拡大変換を行う第二の構成要素と、を含む。

【0056】 本発明の他のハッシュ値生成方法および装置では、ハッシュ値生成の過程において、出力の長さを入力長さより長くし、かつ、入力値が異なれば出力値も必ず異なるような単射拡大変換を行っている。このため、衝突回避性の高い、すなわち安全なハッシュ値を生成することができる。

【0057】 ここで、対象データは、単射拡大変換でのパラメータとして用いる初期値と混合させてから、第一の構成要素に入力するようにしてもよい。このようにすることで、異なった初期値に対して同じハッシュ値が導かれるといったような初期値の衝突が、起こる確率を低くすることができる。

【0058】 また、第一の構成要素に入力した対象データを、再度、第一の構成要素に入力するようにしてもよい。このようにすることで、異なったメッセージに対して同じハッシュ値が導かれるといったようなメッセージの衝突が、起こる確率を低くすることができる。

【0059】 また、本発明のデータ暗号化方法および装置は、一定長のデータを暗号変換して一定長の暗号化データを出力するものであり、対象データを換字および/または転置変換を行う第一の構成要素と、前記第一の構成要素の結果得られたデータに対して、結果が当該データのデータ長よりも長くなるような乗算を行う第二の構成要素と、前記第二の構成要素の結果得られたデータを、少なくとも2つのブロックに分割する第三の構成要素と、前記第三の構成要素の結果得られた少なくとも2つのブロック各々に対して、換字および/または転置変換を行う第四の構成要素と、含む。

【0060】 この構成においても、データ暗号化の過程において、入力値に対して出力値の長さが長くなるような乗算を行っているので、データの攪乱を効率よく行うことができる。

【0061】 さらに、本発明の他のデータ暗号化方法および装置は、公開鍵を用いて平文を暗号化する公開鍵暗号を用いたものであり、第一の公開鍵を変換することで得られたデータをパラメータとして、平文に対して暗号化変換を行う第一の構成要素と、少なくとも1つの第二の公開鍵に応じたデータと、前記第一の公開鍵を変換することで得られたデータとの間において、前記第二の公開鍵に応じたデータが分かれば、前記第一の公開鍵を変

換することによって得られたデータを、直接あるいは間接に求めることができる関係式を満たすデータ値を生成する第二の構成要素と、を含み送信すべき暗号化データとして、前記第一の処理の結果得られたデータに、前記第二の処理の結果得られたデータ値を付与したことを特徴とする。

【0062】また、このデータ暗号化方法および装置と対になるデータ復号化方法および装置は、第二の公開鍵に応じたデータを、当該第二の公開鍵と対になる秘密鍵から求める第三の構成要素と、暗号文に付加されたデータ値と、前記第三の構成要素で求めたデータとを基に、第一の公開鍵を変換することによって得られたデータを求める第四の構成要素と、前記第四の構成要素で求めたデータをパラメータとして、暗号文を復号する第五の構成要素と、を含む。

【0063】この構成によれば、第二の公開鍵と対になる秘密鍵を有する者は、単独あるいは他の第二の公開鍵を有する者と協力することで、第二の構成要素で生成されたデータ値から、第一の公開鍵を変換することによって得られたデータを得ることができる。

【0064】したがって、第一の公開鍵と対の秘密鍵を有する者は勿論のこと、第二の公開鍵と対になる秘密鍵を有する者であっても、データの復号が可能となる。

【0065】このことは、送信側にとっては、同じ電子メールを複数の宛先に暗号化して送信しようとする場合に、各宛先毎に、当該宛先各々から予め配布された公開鍵を用い、電子メールの暗号化処理を一行なう必要がなくなることを意味する。

【0066】

【発明の実施の形態】以下に、本発明の第一実施形態について説明する。

【0067】図1は本発明の第一実施形態であるハッシュ値生成装置の機能構成を示す図である。このハッシュ値生成装置は、たとえば、パーソナルコンピュータやICカードなどのマイクロプロセッサを備えた情報処理装置において、マイクロプロセッサに所定のプログラムを実行させることで実現可能である。また、1つのLSIで実現させることも可能である。なお、このハッシュ値生成装置は、電子捺印や電子メールなどのデータ暗号化に用いることができる。

【0068】図1において、圧縮しようとしているメッセージ3001が、ハッシュ値生成装置101に入力されると、まず、データ拡大部102において、以下の処理が行われる。

【0069】混合処理部103において、入力されたメッセージ3001と初期値との混合処理が行われる。この混合処理については後述する。

【0070】伸長処理部104において、混合処理部103で得た混合データのLブロック毎K回繰り返し伸長処理が行われる。このLブロック毎K回繰り返し伸長

処理については後述する。

【0071】上記の処理によりメッセージ3001の拡大データ107を生成する。

【0072】データ拡大部102で生成された拡大データ107は、第1区分E₁108、第2区分E₂109、・・・というように、64ビット毎のフレーム（ブロックのかたまり）に分割されて、順次、単射拡大部105に入力される。

【0073】単射拡大部105は、256ビットの初期値110をパラメータとして、第1区分E₁108に対して、換字・転置を行いながら単射拡大処理（これについては後述する）を施す。これにより、256ビットの第1番目の中間出力を算出する。

【0074】次に、その第1番目の中間出力をパラメータ（初期値110の代わり）にして、第2区分E₂109に対して、換字・転置を行いながら単射拡大処理を施す。これにより、256ビットの第2番目の中間出力を算出する。

【0075】上記の処理を最後の区分のフレームが入力されるまで繰り返すことで、最後に算出された256ビットの中間出力を、ハッシュ値Hash111として用いる。

【0076】次に、図1に示すハッシュ値生成装置101の各部での処理について、詳述する。

【0077】まず、データ拡大部102の混合処理部103での処理について説明する。

【0078】混合処理部103は、メッセージ3001および初期値110を各々複数のデータブロックに分割して、両者を混ぜ合わせるといった処理を行う。

【0079】図2は、混合処理部103での処理の一例を説明するための図である。

【0080】ここで、初期値110は、4個の64ビットデータブロックI₁201、I₂202、I₃203、I₄204がこの順番に連なって構成されたものとする。

【0081】まず、パディング処理部220において、メッセージ3001の長さに初期値110の長さ（256ビット）を加えた値が、L×64の整数倍となるように、メッセージ3001を加工する。

【0082】ここで、Lとは、後述する伸長処理部104において行われるLブロック毎K回繰り返し伸長処理によって定義される値である。また、64とは、単射拡大部105に入力されるフレーム（1つの区分のデータ）のビット長である。

【0083】パディング処理部220は、具体的には、以下のようにして、メッセージ3001を加工する。

【0084】メッセージ3001の長さに初期値110の長さ（256ビット）を加えた値が、L×64の整数倍である場合、メッセージ3001の後部に、ビット「11」とL×64-2個のビット「0101・・・」

とを、この順番で接続する。

【0085】メッセージ3001の長さに初期値110の長さ(256ビット)を加えた値が、 $L \times 64$ ビットの整数倍でない場合、メッセージ3001の後部に、ビット「11」と、0個～ $L \times 64 - 1$ 個の間でいくつかのビット「0101」とを、この順番に接続する。このようにすることで、メッセージ3001全体の長さが、 $L \times 64$ の整数倍となるようにする。

【0086】次に、パディング処理部220にて、全体の長さが $L \times 64$ の整数倍となるように加工されたメッセージ3001は、 N 個の64ビットデータブロック M_1205 、 M_2206 、 M_3207 、 M_4208 、 M_5209 、・・・がこの順番に連なったデータ216に変換される。

【0087】その後、処理部217において、この N 個の64ビットデータブロックと、初期値110を構成する4個の64ビットデータブロックとを混ぜ合わせる。具体的には、図2に示すように、

「 $M_1 \rightarrow D_1$ 、 $I_1 \rightarrow D_2$ 、 $M_2 \rightarrow D_3$ 、 $I_2 \rightarrow D_4$ 、 $M_3 \rightarrow D_5$ 、 $I_3 \rightarrow D_6$ 、 $M_4 \rightarrow D_7$ 、 $I_4 \rightarrow D_8$ 、 $M_5 \rightarrow D_9$ 、 $M_6 \rightarrow D_{10}$ 、 $M_7 \rightarrow D_{11}$ 、・・・」

といったデータの置き換えが行われる。

【0088】この置き換えた結果としての $N+4$ 個の64ビットデータブロック D_1210 、 D_2311 、 D_3212 、 D_4213 、 D_5214 、・・・がこの順番で連なったものが、中間拡大データ215として出力される。

【0089】この中間拡大データ215の長さは、 $L \times 64$ ビットの整数倍となる。

【0090】次に、データ拡大部102の伸長処理部104での処理について説明する。

【0091】伸長処理部104は、中間拡大データ215の伸長処理を行う。

【0092】図3は、伸長処理部104での処理の一例を説明するためのフロー図である。

【0093】ここでは、中間拡大データ215の L ブロック分をコピーし、これをコピーしたブロックのうちの最後のブロックの後に追加するといった動作を、 L ブロック毎に K 回繰り返して行っている。本実施形態では、この処理を L ブロック毎 K 回繰り返し伸長処理と称している。

【0094】まず、ステップ302において、混合処理部103で求めた中間拡大データ215を構成する $N+4$ 個の64ビットデータブロック D_1210 、 D_2311 、 D_3212 、 D_4213 、 D_5214 、・・・を入力する。

【0095】次に、ステップ303において、 $i=1$ 、 $j=0$ を設定する。

【0096】次に、ステップ304において、次式により m を求める。

【0097】 $m = (i - (i \pmod{L \cdot K})) / K +$

$((i-1) \pmod{L}) + 1$

ここで、 \pmod{X} とは、 X で割った余りをとるという処理を示す。たとえば、 $5 \pmod{2} = 1$ である。

【0098】次に、ステップ305において、中間拡大データ215を構成する $N+4$ 個の64ビットデータブロックのうち、 m 番目のデータブロック D_m を、上述した拡大データ107を構成するフレームのうちの i 番目のフレーム E_i に設定する。

【0099】次に、ステップ306において、ステップ305で設定したフレーム E_i を出力する。

【0100】次に、ステップ307において、 m 番目のデータブロック D_m が、中間拡大データ215を構成する $N+4$ 個の64ビットデータブロックのうちの最後のブロックに相当するか否かを判断する。最後のブロックである場合は、ステップ308へ移行し、そうでない場合は、ステップ310へ移行する。

【0101】ステップ308では、 j の値を1つインクリメント($j = j + 1$)し、ステップ309へ移行する。

【0102】ステップ309では、 j が K より大きい(否)かを判断する($j > K?$)。 j が K より大きい場合はこのフローを終了し、 j が K 以下の場合はステップ310へ移行する。

【0103】ステップ310では、 i の値を1つインクリメント($i = i + 1$)し、その後、ステップ304へ戻る。

【0104】上記のフローを実行することで、中間拡大データ215に対して、上記説明した L ブロック毎 K 回繰り返し伸長処理が施され、結果として、図1に示す拡大データ107を構成するフレーム E_1108 、 E_2109 、・・・が順次出力される。なお、拡大データ107は、中間拡大データ215の K 倍になる。

【0105】次に、単射拡大部105での処理について説明する。

【0106】単射拡大部105は、当該単射拡大部105に入力される拡大データ107に対して換字・転置処理を行うという点では、従来の「ブロック暗号を利用したハッシュ関数」や「専用ハッシュ関数」と同じである。

【0107】しかしながら、本実施形態の単射拡大部105は、当該単射拡大部105に入力された、拡大データ107を構成するフレーム各々について、入力値が異なれば出力値も必ず異なり(単射)、且つ入力値の長さよりも出力値の長さの方が長くなる(拡大)ように、入力されたフレームを変換する処理を行う点で、従来のハッシュ関数と異なる。なお、本実施形態では、この処理を単射拡大処理と称している。

【0108】図4は、単射拡大部105での処理の一例を説明するためのフロー図である。

【0109】まず、ステップ402において、256ビ

ットの初期値110を入力する。そして、これをHに設定する。

【0110】次に、ステップ403において、 $q=1$ に設定する。

【0111】次に、ステップ404において、図1に示す拡大データ107のうち、 q 番目のフレーム E_q を入力する。

【0112】次に、ステップ405において、フレーム E_q に対し、Hをパラメータとして、その長さを64ビットから96ビットへ拡大するような単射拡大処理を施す。

【0113】次に、ステップ406において、ステップ405で得たデータに対し、Hをパラメータとして、その長さを96ビットから128ビットへ拡大するような単射拡大処理を施す。

【0114】次に、ステップ407において、ステップ406で得たデータに対し、Hをパラメータとして、その長さを128ビットから256ビットへ拡大するような単射拡大処理を施す。そして、得た256ビットのデータをHに設定する。

【0115】次に、ステップ408では、 q 番目のフレーム E_q が、拡大データ107を構成するフレーム E_1 、 E_2 ・・・のうちの最後のフレームに相当するか否かを判断する。最後のフレームである場合は、ステップ410へ移行し、そうでない場合は、ステップ409へ移行する。

【0116】ステップ409では、 q の値を1つインクリメント($q=q+1$)し、その後、ステップ404へ戻る。

【0117】ステップ410では、ステップ407で設定した256ビットのデータHが、拡大データ107を構成するフレームのうちの最後のフレームに対するものであるので、このHをハッシュ値Hash111として出力する。

【0118】次に、図4に示すステップ405～407での単射拡大処理について説明する。

【0119】まず、図4に示すステップ405での単射拡大処理(64ビット→96ビットの単射拡大)について説明する。

【0120】図5は、図4に示すステップ405における64ビットデータから96ビットデータへの単射拡大処理の一例を説明するためのフロー図である。

【0121】まず、ステップ502において、図4に示すステップ404で入力された64ビットのフレーム E_q を、上位32ビットのデータ X_1 と下位32ビットのデータ Y_1 とに分割する。

【0122】また、256ビットのデータH(図4において、ステップ404で入力されたフレームが1番目のフレーム E_1 である場合は、ステップ402で設定した値、2番目以降のフレーム E_2 ・・・である場合は、直

前に実施されたステップ407で設定した値)を、先頭から32ビット毎に分割して、8個の32ビットデータ H_1 、 H_2 、 H_3 、・・・ H_8 を得る。

【0123】次に、ステップ503において、次式で示される処理を行うことで、 X_2 、 Y_2 を生成する。

$$\begin{aligned} \text{【0124】} X_2 &= X_1 + (Y_1 + H_1)^2 \\ Y_2 &= Y_1 \end{aligned}$$

この結果、 X_2 は64ビット、 Y_2 は32ビットのデータとなる。なお、図5のステップ503において、太線の矢印は64ビットデータの流れを、そして、細線の矢印は、32ビットデータの流れを示している。

【0125】次に、ステップ504において、 X_2 、 Y_2 を出力し、その後、このフローを終了する。

【0126】上記のフローにより、32ビットのデータ X_1 および32ビットのデータ Y_1 からなるフレーム E_q を、64ビットのデータ X_2 および32ビットのデータ Y_2 の合計96ビットのデータに拡大することができ、且つ、 X_2 、 Y_2 が与えられたとき、 $X_1 = X_2 - (Y_2 + H_1)^2$ 、および、 $Y_1 = Y_2$ から、 X_1 、 Y_1 が一意的に定まる関係、すなわち単射とすることができる。

【0127】したがって、上記のフローにより、64ビットデータから96ビットデータへの単射拡大を行うことができる。

【0128】ただし、図4に示すステップ405での処理は、図5に示すものに限定されるものではなく、64ビットデータから96ビットデータへの単射拡大処理を行うものであればよい。

【0129】次に、図4に示すステップ406での単射拡大処理(96ビット→128ビットの単射拡大)について説明する。

【0130】図6は、図4に示すステップ406における96ビットデータから128ビットデータへの単射拡大処理の一例を説明するためのフロー図である。

【0131】まず、ステップ602において、図5に示すフローで生成した64ビットデータ X_2 および32ビットデータ Y_2 を入力する。また、図5に示すステップ502により256ビットのデータHを分割することで生成した、8個の32ビットデータ H_1 、 H_2 、 H_3 、・・・ H_8 のうち、 H_2 、 H_3 、 H_4 、 H_5 、 H_6 、 H_7 、 H_8 を入力する。

【0132】次に、ステップ603において、64ビットデータ X_2 を、上位32ビットのデータ X_H と下位32ビットのデータ X_L とに分割する。

【0133】次に、ステップ604において、次式で示される処理を順番に行うことで X_3 、 Y_3 を生成する。

$$\begin{aligned} \text{【0134】} A &= X_L \text{ eor } H_2 \\ B &= X_H + H_3 + 1 \\ C &= A \cdot B \\ C' &= (C \text{ eor } Y_2) + (H_4 \parallel H_5) + 1 \\ C''_H \parallel C''_L &= C' \end{aligned}$$

$D = \text{rot}_5 (C \ll L) \text{ eor } H_6$
 $E = \text{rot}_{12} (C \ll H) \div H_7 + 1$
 $F = (D \parallel E)$
 $G = X_2 \div F + 1$
 $X_3 = C \ll G \div (H_8 \parallel H_1)$
 $Y_3 = G$

ここで、eor は、ビット毎の排他的論理和を示す。たとえば、 $110010 \text{ eor } 011001 = 101011$ である。また、+ は加算を示す。ただし、最上位ビットの計算において桁上げが生じた場合は、桁上げ部分を無視する。たとえば、 $101101 + 100100 = 010001$ となる。

【0135】また、 \parallel はデータの結合を示す。たとえば、 $111111 \parallel 000000 = 111111000000$ である。また、 $\text{rot}_T (U)$ は、数値データ U を上位側へ T ビット巡回シフトすることで得られるデータを示す。たとえば、 $\text{rot}_2 (110000) = 000011$ となる。ここで、数値データの左側が上位側である。

【0136】上記の式で示される処理を順次行うことにより、 X_3 および Y_3 は、各々 64 ビットのデータとなる。なお、図 6 のステップ 604 において、太線の矢印は 64 ビットデータの流れを、そして、細線の矢印は、32 ビットデータの流れを示している。

【0137】次に、ステップ 605 において、 X_3 、 Y_3 を出力し、その後、このフローを終了する。

【0138】上記のフローにより、64 ビットのデータ X_2 および 32 ビットのデータ Y_2 の合計 96 ビットのデータを、64 ビットのデータ X_3 および 64 ビットのデータ Y_3 の合計 128 ビットのデータに拡大することができ、且つ、 X_3 、 Y_3 が与えられたとき、 X_2 、 Y_2 が一意的に定まる関係、すなわち単射とすることができる。

【0139】したがって、上記のフローにより、96 ビットデータから 128 ビットデータへの単射拡大を行うことができる。

【0140】ただし、図 4 に示すステップ 406 での処理は、図 6 に示すものに限定されるものではなく、96 ビットデータから 128 ビットデータへの単射拡大処理を行うものであればよい。

【0141】次に、図 4 に示すステップ 407 での単射拡大処理 (128 ビット → 256 ビットの単射拡大) について説明する。

【0142】図 7 は、図 4 に示すステップ 407 における 128 ビットデータから 256 ビットデータへの単射拡大処理の一例を説明するためのフロー図である。

【0143】まず、ステップ 702 において、図 6 に示すフローで生成した 64 ビットデータ X_3 、 Y_3 を入力する。また、図 5 に示すステップ 502 により 256 ビットのデータ H を分割することで生成した 8 個の 32 ビットデータ H_1 、 H_2 、 H_3 、 H_4 、 H_5 、 H_6 、 H_7 、 H_8 を入力する。

【0144】次に、ステップ 703 において、64 ビットデータ X_3 を、上位 32 ビットのデータ X_H と下位 32 ビットのデータ X_L とに分割する。

【0145】次に、ステップ 704 において、64 ビットデータ Y_3 を、上位 32 ビットのデータ Y_H と下位 32 ビットのデータ Y_L とに分割する。

【0146】次に、ステップ 705 において、次式で示される処理を順番に行うことで、換字・転置を繰り返しながら、 K_1 、 K_2 、 \dots 、 K_8 を生成する。

10 【0147】 $K_1 \parallel K_3 = (H_8 \parallel H_6) \div (X_H \parallel X_L) + ((X_H \text{ eor } Y_H) \parallel (H_L \text{ eor } Y_L))$
 $K_2 \parallel K_4 = (H_7 \parallel H_5) \text{ eor } ((H_8 \parallel H_6) \div (X_H \parallel X_L))$
 $K_5 \parallel K_7 = (H_4 \parallel H_2) + (Y_H \parallel Y_L) + ((X_L \text{ eor } Y_H) \parallel (X_H \text{ eor } Y_L))$
 $K_6 \parallel K_8 = (H_3 \parallel H_1) \text{ eor } ((H_4 \parallel H_2) + (Y_H \parallel Y_L))$

この結果、 K_1 、 K_2 、 K_3 、 K_4 、 K_5 、 K_6 、 K_7 、 K_8 は、各々 32 ビットのデータとなる。なお、図 7 のステップ 705 において、細線の矢印は、32 ビットデータの流れを示している。

【0148】次に、ステップ 706 において、 K_1 、 K_2 、 K_3 、 K_4 、 K_5 、 K_6 、 K_7 、 K_8 をこの順序で結合して、 H を生成する ($K_1 \parallel K_2 \parallel K_3 \parallel K_4 \parallel K_5 \parallel K_6 \parallel K_7 \parallel K_8 \rightarrow H$)。これにより、 H は 32 ビット \times 8 = 256 ビットのデータとなる。

【0149】ステップ 708 では、ステップ 706 で生成した H を出力し、その後、このフローを終了する。

30 【0150】上記のフローにより、64 ビットのデータ X_3 および Y_3 の合計 128 ビットのデータを、256 ビットのデータ H に拡大することができ、且つ、 H が与えられたとき、 X_3 、 Y_3 が一意的に定まる関係、すなわち単射とすることができる。

【0151】したがって、上記のフローにより、128 ビットデータから 256 ビットデータへの単射拡大を行うことができる。

40 【0152】ただし、図 4 に示すステップ 407 での処理は、図 7 に示すものに限定されるものではなく、128 ビットデータから 256 ビットデータへの単射拡大処理を行うものであればよい。

【0153】上記の第一実施形態では、入力されたフレーム E_q と、初期値 110 あるいは単射拡大部 105 の出力である H とを用いて、ハッシュ値を生成する過程において、換字・転置処理を行う単射拡大部 105 において、32 ビットデータ同士の乗算処理を行っている (図 5 に示すステップ 503 での処理 $X_2 = X_1 + (Y_1 + H_1)^2$ 、および図 6 に示すステップ 604 での処理 $C = A \cdot B$)。併せて、32 ビットデータに対する巡回シフト計算をも行っている (図 6 に示すステップ 604 での

50 処理 $D = \text{rot}_5 (C \ll L) \text{ eor } H_6$ 、 $E = \text{rot}_{12} (C \ll H)$)

+H7+1)。

【0154】このようにすることで、従来の「ブロック暗号を利用したハッシュ関数」や「専用ハッシュ関数」に比べて、データ攪乱度の高いハッシュ値を、迅速に生成することができる。

【0155】すなわち、32ビットデータ同士の乗算(32ビット×32ビット→64ビット)は、出力される64ビットデータの各ビットが、すべての入力ビットの影響を受ける。このため、データの攪乱度が高いので、効率よく換字処理を行うことができる。

【0156】なお、現在、主にパーソナルコンピュータ用のマイクロプロセッサとして普及しているインテル社製の100MHzのペンティアムプロセッサならば、1秒間に1千万回の積演算(乗算)を行うことができる。これは、1980年代中期に発表されたモトローラ社製の20MHzの68020プロセッサでは1秒間に50万回程度しか積演算を行うことができなかったのに対して、およそ20倍の高速化となっている。

【0157】また、32ビットデータの巡回シフト計算も、効率よく換字処理を行うという点で有効である。

【0158】マイクロプロセッサによる演算処理では、一回の処理で巡回シフト計算、すなわち32ビットデータの転置処理を実現することができるが、最近のマイクロプロセッサ、たとえば、インテル社製のペンティアムプロセッサによれば、1サイクルでこの巡回シフトを完了する。インテル社製の100MHzのペンティアムプロセッサならば、1秒間に1億回の巡回シフト計算処理を行うことができる。これは、1980年代中期に発表されたモトローラ社製の20MHzの68020プロセッサでは1秒間に250万回程度しか巡回シフトを行うことができなかったのに対して、およそ40倍の高速化となっている。

【0159】このように、本発明の第一実施形態は、換字・転置処理を行うに際して、最近の技術革新により特に有利になっているマイクロプロセッサの基本演算を使用することで、データ攪乱度の高いハッシュ値の生成を迅速に行うようにしている。

【0160】ところで、従来の「ブロック暗号を利用したハッシュ関数」や「専用ハッシュ関数」では、通常、32ビットデータ同士の加算を行うことで、換字処理を実現している。この32ビットデータ同士の加算処理は、インテル社製の100MHzのペンティアムプロセッサならば、1秒間に1億回実行することができる。これは、1980年代中期に発表されたモトローラ社製の20MHzの68020プロセッサでは1秒間に1千万回程度しか加算を行うことができなかったのに対して、およそ10倍程度しか高速化されていない。

【0161】乗算処理は、32回分の加算処理と、32回分の巡回シフト処理とを行ったのと同じデータ攪乱効果がある。このことを考慮すれば、インテル社製のペン

ティアムプロセッサが主流となっている現在においては、加算処理よりも乗算処理を用いる有利性が、さらに増したといえる。

【0162】また、本発明の第一実施形態では、入力されたフレームE_qと、初期値110あるいは単射拡大部105の出力であるHとを用いて、ハッシュ値を生成する過程において、換字・転置処理を行う単射拡大部105において、当該単射拡大部105に入力された、拡大データ107を構成するフレーム各々について、入力値が異なれば出力値も必ず異なり(単射)、且つ入力値の長さよりも出力値の長さの方が長くなる(拡大)ように、入力されたフレームを変換する処理を行っている。

【0163】これにより、衝突回避性の高い、すなわち安全なハッシュ値を生成することができる。

【0164】すなわち、従来のハッシュ関数では、換字・転置繰り返し処理へ入力されるメッセージの区分(フレーム)の長さ、出力される中間出力の長さとを比べた場合、中間出力の長さは、入力されるフレームの長さと等しいか(「ブロック暗号を利用したハッシュ関数」)、あるいは短かった(「専用ハッシュ関数」)。

【0165】これに対し、本発明の第一実施形態では、上記の単射拡大処理により、中間出力の長さ(256ビット)が、入力されるフレームの長さ(64ビット)より長くなる。したがって、MD5で問題となったような、換字・転置の繰り返し処理におけるメッセージの衝突回避を比較的容易に実現できる。

【0166】また、本発明の第一実施形態では、図1に示すように、初期値110を、単射拡大部105へ入力する最初のパラメータとしてのみならず、メッセージ3001を単射拡大部105へ入力する前段階の処理として、当該メッセージ3001を拡大して拡大データ107を生成するのにも用いている。

【0167】このようにすることで、図27に示すように、初期値3004を換字・転置繰り返し処理3005へ入力する最初のパラメータとしてのみ用いていた従来のハッシュ関数に比べ、異なった初期値に対して同じハッシュ値が導かれるといったような、初期値の衝突が起こる確率を、低減させることができる。

【0168】さらに、本発明の第一実施形態では、図1に示すように、メッセージ3001を単射拡大部105へ入力する前段階の処理として、メッセージを分割して複数のブロックを生成し、この生成した複数のブロックの一部あるいは全部をコピーして、元の複数のブロックに混ぜるといった処理(本実施形態では、Lブロック毎K回繰り返し伸長処理と称している)を行っている。

【0169】このようにすることで、図27に示すように、メッセージ3001を換字・転置繰り返し処理3005へ入力する前段階の処理として、当該メッセージ3001を分割して、複数の区分P₁、P₂・・・を生成するのみであった従来のハッシュ関数に比べ、異なったメ

ッセージに対して同じハッシュ値が導かれるといったような、メッセージの衝突が起こる確率を、低減させることができる。

【0170】なお、上記の第一実施形態では、換字・転置を行う単射拡大部256での32ビットデータ乗算処理として、図5に示すステップ503の $X_2 = X_1 + (Y_1 + H_1)^2$ で示される処理と、図6に示すステップ604の $C = A \cdot B$ で示される処理と、を行うものについて説明した。しかしながら、本発明でいう乗算処理がこれ等の式に限定されるものでないことは、当然のことである。

【0171】同様に、本発明でいう巡回シフト計算処理も、図6に示すステップ604の $D = \text{rot}_5(C \ll L) \oplus H_6$ 、 $E = \text{rot}_{12}(C \ll H) + H_7 + 1$ で示される処理に限定されるものではない。

【0172】また、上記の実施形態では、単射拡大部105へ入力される拡大データ107のフレームE1、E2、・・・を64ビットとし、単射拡大部105から出力される中間拡大データを256ビット（したがって、ハッシュ値は256ビット）としたものについて説明した。しかしながら、本発明はこれに限定されるものではない。

【0173】以下に、本発明の第一実施形態の変形例として、単射拡大部へ入力されるフレームを64ビットとし、単射拡大部からの中間出力を80ビット（したがって、ハッシュ値は80ビット）としたものについて説明する。

【0174】図8は本発明の第一実施形態の変形例であるハッシュ値生成装置の機能構成を示す図である。ここで、図1に示すハッシュ値生成装置101と同じ機能を有するものには、同じ符号を付している。

【0175】図8に示すハッシュ値生成装置101aが図1に示すハッシュ値生成装置101と異なる点は、256ビットの初期値110に代えて80ビットの初期値802を用いた点、混合処理部103に代えて混合処理部801を用いた点、および、単射拡大部105に代えて単射拡大部803を用いた点である。その他の構成は、図1に示すものと同様である。

【0176】図8に示すハッシュ値生成装置101aでは、80ビットのハッシュ値Hash804を生成する。

【0177】混合処理部801は、メッセージ2501および初期値802各々を複数のデータブロックに分割して、両者を混ぜ合わせるといった処理を行う点で、図1に示す混合処理部103と同様である。ただし、80ビットの初期値802を用いているため、具体的な処理が異なってくる。

【0178】図9は、混合処理部801での処理の一例を説明するための図である。

【0179】ここで、初期値802は、64ビットのデータブロックI1901と、16ビットのデータブロッ

クI2902とが、この順番で連なって構成されたものとする。

【0180】まず、パディング処理部220において、メッセージ2501の長さに初期値802の長さ（80ビット）を加えた値が、 $L \times 64$ の整数倍となるように、メッセージ2501を加工する。この処理は、図2に示す混合処理部103でのパディング処理220と同様である。

【0181】次に、パディング処理部220にて、全体の長さが $L \times 64$ の整数倍となるように加工されたメッセージ2501は、N個の64ビットデータブロックM1205、M2206、M3207、M4208、M5209、・・・がこの順番に連なったデータ216に変換される。

【0182】その後、処理部903において、このN個の64ビットデータブロックと、初期値802を構成する64ビットのデータブロックI1901および16ビットのデータブロックI2902と、を混ぜ合わせる。

具体的には、図9に示すように、

「M1→D1、I1→D2、M2→D3、I2||I2||I2||I2→D4、M3→D5、M4→D6、M5→D7、M6→D8、M7→D9、・・・」

といったデータの置き換えが行われる。

【0183】この置き換えた結果としてのN+2個の64ビットデータブロックD1210、D2311、D3212、D4213、D5214、・・・がこの順番で連なったものが、中間拡大データ215として出力される。

【0184】この中間拡大データ215の長さは、 $L \times 64$ ビットの整数倍となる。

【0185】単射拡大部803は、当該単射拡大部803に入力される拡大データ107に対して、換字・転置処理を行いながら単射拡大を行うという点で、図1に示す単射拡大部105と同様である。ただし、単射拡大部803からの出力（中間出力）を80ビットのデータとするために、具体的な処理が異なってくる。

【0186】図10は単射拡大部803での処理の一例を説明するためのフロー図である。

【0187】まず、ステップ1002において、80ビットの初期値802を入力する。そして、これをHに設定する。

【0188】次に、ステップ1003において、 $q = 1$ に設定する。

【0189】次に、ステップ1004において、図8に示す拡大データ107のうち、q番目のフレームEqを入力する。

【0190】次に、ステップ1005において、フレームEqに対し、Hをパラメータとして単射処理を施す。

【0191】次に、ステップ1006において、ステップ1005で得たデータに対し、Hをパラメータとして、その長さを64ビットから80ビットへ拡大するよ

うな単射拡大処理を施す。そして、得た80ビットのデータをHに設定する。

【0192】次に、ステップ1007では、q番目のフレーム E_q が、拡大データ107を構成するフレーム E_1, E_2, \dots のうちの最後のフレームに相当するか否かを判断する。最後のフレームである場合は、ステップ1009へ移行し、そうでない場合は、ステップ1008へ移行する。

【0193】ステップ1008では、qの値を1つインクリメント($q = q + 1$)し、その後、ステップ1004へ戻る。

【0194】ステップ1009では、ステップ1006で設定した80ビットのデータHが、拡大データ107を構成するフレームのうちの最後のフレームに対するものであるため、このHをハッシュ値Hash804として出力する。

【0195】次に、図10に示すステップ1005での単射処理、およびステップ1006での単射拡大処理について説明する。

【0196】まず、図10に示すステップ1005での単射処理(64ビット→64ビットの単射)について説明する。

【0197】図11は、図10に示すステップ1005における64ビットデータから64ビットデータへの単射処理の一例を説明するためのフロー図である。

【0198】まず、ステップ1102において、図10に示すステップ1004で入力された64ビットのフレーム E_q を、上位32ビットのデータ X_1 と下位32ビットのデータ Y_1 とに分割する。

【0199】また、80ビットのデータH(図10において、ステップ1004で入力されたフレームが1番目のフレーム E_1 である場合は、ステップ1002で設定した値、2番目以降のフレーム E_2, \dots である場合は、直前に実施されたステップ1006で設定した値)を、先頭から32ビットデータ H_1 、32ビットデータ H_2 、および16ビットデータ H_3 に分割する。

【0200】次に、ステップ1103において、次式で示される処理を行うことで、 X_2, Y_2 を生成する。

$$\begin{aligned} \text{【0201】 } X_2 &= X_1 + (Y_1 + H_1)^2 \pmod{2^{32}} \\ Y_2 &= Y_1 \end{aligned}$$

この結果、 X_2, Y_2 は、ともに32ビットのデータとなる。なお、図11のステップ1103において、細線の矢印は32ビットデータの流れを示している。

【0202】次に、ステップ1104において、 X_2, Y_2 を出力し、その後、このフローを終了する。

【0203】上記のフローにより、32ビットのデータ X_1 および32ビットのデータ X_1 からなるフレーム E_q を、32ビットのデータ X_2 および32ビットのデータ Y_2 の合計64ビットのデータに変換することができる。また、 X_2, Y_2 が与えられたとき、 $X_1 = X_2 - (Y$

$2 + H_1)^2 \pmod{2^{48}}$ 、および、 $Y_1 = Y_2$ から、 X_1, Y_1 が一意的に定まる関係、すなわち単射とすることができる。

【0204】したがって、上記のフローにより、64ビットデータから64ビットデータへの単射を行うことができる。

【0205】ただし、図10に示すステップ1005での処理は、図11に示すものに限定されるものではなく、64ビットデータから64ビットデータへの単射処理を行うものであればよい。

【0206】次に、図10に示すステップ1006での単射拡大処理(64ビット→80ビットの単射拡大)について説明する。

【0207】図12は、図10に示すステップ1006における64ビットデータから80ビットデータへの単射拡大処理の一例を説明するためのフロー図である。

【0208】まず、ステップ1202において、図11に示すフローで生成した32ビットデータ X_2, Y_2 を入力する。また、図11に示すステップ1102により80ビットのデータHを分割することで生成した、32ビットデータ H_1, H_2 と、16ビットデータ H_3 とを入力する。

【0209】次に、ステップ1203において、次式で示される処理を順番に行うことで X_3, Y_3 を生成する。

$$\begin{aligned} \text{【0210】 } A &= \text{rot5}(X_2 + H_2 + 1) \\ B &= A + X_2 \text{ eor } (H_3 \parallel H_3) + 1 \\ C &= \text{rot13}(B + H_1) \\ D &= C + (B \text{ or } H_2) + 1 \\ E &= D^2 + Y_2 \pmod{2^{48}} \\ F &= X_2 + H_3 + E \pmod{2^{32}} \\ X_3 &= E + F + (H_1 \parallel H_2) \pmod{2^{48}} \\ Y_3 &= F \end{aligned}$$

ここで、or は、ビット毎の論理和を示す。

【0211】上記の式で示される処理を順次行うことにより、 X_3 は48ビットのデータ、そして、 Y_3 は32ビットのデータとなる。なお、図12のステップ1203において、太線の矢印は48ビットデータの流れを、そして、細線の矢印は、32ビットデータの流れを示している。

【0212】次に、ステップ1204において、 X_3, Y_3 を出力し、その後、このフローを終了する。

【0213】上記のフローにより、32ビットデータ X_2 および32ビットデータ Y_2 の合計64ビットのデータを、48ビットのデータ X_3 および32ビットのデータ Y_3 の合計80ビットのデータに拡大することができ、且つ、 X_3, Y_3 が与えられたとき、 X_2, Y_2 が一意的に定まる関係、すなわち単射とすることができる。

【0214】したがって、上記のフローにより、64ビットデータから80ビットデータへの単射拡大を行うことができる。

【0215】ただし、図10に示すステップ1006での処理は、図12に示すものに限定されるものではなく、64ビットデータから80ビットデータへの単射拡大処理を行うものであればよい。

【0216】次に、本発明の第二実施形態について説明する。

【0217】図13は本発明の第二実施形態であるデータ暗号化装置の機能構成を示す図である。このデータ暗号化装置は、第一実施形態のハッシュ値生成装置と同様に、パーソナルコンピュータやICカードなどのマイクロプロセッサを備えた情報処理装置において、マイクロプロセッサに所定のプログラムを実行させることで実現可能である。また、1つのLSIで実現させることも可能である。

【0218】図13において、任意長のデータ鍵1302が、データ暗号化装置1311に入力されると、ハッシュ値生成装置1301において、256ビットのシステム鍵1303が初期値として与えられ、当該データ鍵1302に対する256ビットのハッシュ値が生成される。このハッシュ値がワーク鍵1304となる。

【0219】ワーク鍵1304は、8個の32ビットデータ W_1, W_2, \dots, W_8 に分割される。

【0220】ここで、ハッシュ値生成装置1302は、本発明の第一実施形態で説明したものでもよいし、従来の「ブロック暗号を利用したハッシュ関数」や「専用ブロック関数」であってもよい。

【0221】また、暗号化の対象となる128ビットの平文1305が、データ暗号化装置1311に入力されると、当該平文1305は、2個の64ビットデータに分割されて、 π 関数処理部1306に入力される。

【0222】そして、8個の32ビットデータ W_1, W_2, \dots, W_8 を鍵として、2個の64ビットのデータに変換（これについては後述する）された後、 π 関数処理部1307に入力され、 π 関数処理部1306での処理と同じ要領で、2個の64ビットのデータに変換される。

【0223】その後、上記の処理が π 関数処理部1308～1313で順次行われ、これにより、 π 関数処理部1313から2個の64ビットデータが出力される。この2個の64ビットデータが結合され、128ビットの暗号文1310が生成される。

【0224】次に、図13に示す π 関数処理部1306～1313での処理について説明する。

【0225】 π 関数処理1306～1313は、入力された2つの64ビットデータに対し、ワーク鍵1304をパラメータとして、換字・転置処理を行う。ただし、本実施形態の π 関数処理1306～1313は、従来の換字・転置処理で用いられているDESなどの暗号化関数と異なり、32ビット同士の乗算処理、および32ビットの巡回シフト計算を含むようにしている。

【0226】図14は、 π 関数処理1306～1313における処理の一例を説明するためのフロー図である。

【0227】まず、ステップ1402において、2個の64ビットデータ X_1, Y_1 （ π 関数処理1306では、平文128ビットを2つに分割することで得られたデータ、 π 関数処理部1307～1313では、一つ前の π 関数処理部から出力されたデータ）を入力する。また、ワーク鍵1304を構成する8個の32ビットデータ $W_1 \sim W_8$ を入力する。

10 【0228】次に、ステップ1403において、ステップ1402で入力した64ビットデータ X_1 を、上位32ビットのデータ X_H と下位32ビットのデータ X_L とに分割する。

【0229】次に、ステップ1404において、次式で示される処理を順番に行うことで、 X_2, Y_2 を生成する。

$$\text{【0230】 } A = X_L \text{ eor } W_1$$

$$B = X_H + W_2 + 1$$

$$C = A \cdot B$$

$$20 \quad C' = (C \text{ eor } Y_2) + (W_3 \parallel W_4) + 1$$

$$C'_H \parallel C'_L = C'$$

$$D = \text{rot}_5(C'_L) \text{ eor } W_5$$

$$E = \text{rot}_{12}(C'_H) + W_6 + 1$$

$$F = D \parallel E$$

$$G = X_1 + F + 1$$

$$X_2 = C' + G + (W_7 \parallel W_8)$$

$$Y_2 = G$$

上記の式で示される処理を順次行うことにより、 X_2 および Y_2 は、ともに64ビットのデータとなる。なお、

30 図14のステップ1404において、太線の矢印は64ビットデータの流れを、そして、細線の矢印は、32ビットデータの流れを示している。

【0231】次に、ステップ1405において、 X_2, Y_2 を出力し、その後、このフローを終了する。

【0232】上記のフローにより、64ビットのデータ X_1, Y_1 の合計128ビットのデータを、換字・転置を行って、64ビットのデータ X_2, Y_2 の合計128ビットのデータに変換することができる。

40 【0233】また、 $X_1 \parallel Y_1 \rightarrow X_2 \parallel Y_2$ への変換処理を全単射とすることができる。すなわち、出力 $X_2 \parallel Y_2$ から入力 $X_1 \parallel Y_1$ へ逆変換することが可能な関数である π^{-1} 関数が存在する。

【0234】具体的には、次式で示される処理を順次行うことで、 X_2, Y_2 から X_1, Y_1 を求めることができる。

$$\text{【0235】 } G = Y_2$$

$$C' = X_2 - G - (W_7 \parallel W_8)$$

$$C'_H \parallel C'_L = C'$$

$$D = \text{rot}_5(C'_L) \text{ eor } W_5$$

$$50 \quad E = \text{rot}_{12}(C'_H) + W_6 + 1$$

$$F = D \parallel E$$

$$X_1 = G - F - 1$$

$$X_H \parallel X_L = X_2$$

$$A = X_L \text{ eor } W_1$$

$$B = X_H - W_2 - 1$$

$$C = A \cdot B$$

$$Y_1 = (C \cdot (W_3 \cdot W_4) - 1) \text{ eor } C$$

上記の式で示される処理を順次行うことにより、 X_1 および Y_1 を求めることができるので、このデータ暗号化装置1311が生成した暗号文1310は、 π^{-1} 関数を用いた逆変換により、元の平文1305に復号化することができる。

【0236】なお、図13に示す π 関数処理部1306～1313での処理は、図14に示すものに限定されるものではなく、全単射の換字・転置を行うに際して、32ビット同士の乗算処理、および32ビットの巡回シフト計算を含むようにしたものであればよい。

【0237】本実施形態の第二実施形態によれば、ワーク鍵1304を用いて平文1305を暗号化する過程において、換字・転置処理を行う π 関数処理部1306～1313において、32ビットデータ同士の乗算処理を行っている（図14に示すステップ1404での処理 $C = A \cdot B$ ）。併せて、32ビットデータに対する巡回シフト計算をも行っている（図14に示すステップ1404での処理 $D = \text{rot}_5(C \cdot L) \text{ eor } W_5$ 、 $E = \text{rot}_{12}(C \cdot H) + W_6 + 1$ ）。

【0238】このようにすることで、上述したように、換字・転置処理にDESなどの暗号化関数を用いた場合にくらべて、データ攪乱度の高い暗号文を、迅速に生成することができる。

【0239】なお、上記の第二実施形態では、図13に示すように、 π 関数処理部での処理を8回行うもの（ $n = 8$ ）について説明したが、本発明はこれに限定されるものではない。 n の値を外部から与えるようにして、 n を任意の正の整数に変更できるようにしてもよい。

【0240】次に、本発明の第三実施形態について説明する。

【0241】図15は本発明の第三実施形態であるマスキング装置の機能構成を示した図である。ここで、マスキング装置とは、データをマスクする（覆い隠す）ためのデータを生成する装置のことである。マスキング装置で生成されたデータは、たとえばデータを暗号化するための鍵として用いることも可能である。

【0242】本実施形態のデータ暗号化装置は、第一実施形態のハッシュ値生成装置と同様に、パーソナルコンピュータやICカードなどのマイクロプロセッサを備えた情報処理装置において、マイクロプロセッサに所定のプログラムを実行させることで実現可能である。また、1つのLSIで実現させることも可能である。

【0243】図15において、128ビットの共通鍵1

502と、この共通鍵1502をN個繋げて構成された伸長データ1503とが入力されると、ハッシュ値生成装置1509において、乱数生成装置1508で生成された乱数1521と、入力された共通鍵1502とを繋ぎ合わせたデータに対する128ビットのハッシュ値が生成される。このハッシュ値が、伸長データ1503の最初の128ビットデータである第一区分1504を暗号化するためのワーク鍵1524となる。また、乱数生成装置1503で生成された乱数1521が、マスキングデータ1520の最初のデータとなる。

【0244】ここで、ハッシュ値生成装置1302は、本発明の第一実施形態で説明したものでもよいし、従来の「ブロック暗号を利用したハッシュ関数」や「専用ブロック関数」であってもよい。

【0245】一方、 π 関数処理部1513において、ワーク鍵1524の一部をパラメータとして、伸長データ1503の第一区分1504が換字・転置されて、128ビットのデータに変換される。その後、 π 関数処理部1514において、ワーク鍵1524の一部をパラメータとして、 π 関数処理部1513で生成された128ビットのデータが換字・転置されて、128ビットのデータに変換される。このデータが、マスキングデータ1520の2番目のデータ g_1 1522となる。また、 π 関数処理部1513で生成された128ビットのデータが、伸長データ1503の2番目の128ビットデータである第二区分1505を暗号化するためのワーク鍵1525となる。

【0246】また、 π 関数処理部1518において、ワーク鍵1525の一部をパラメータとして、伸長データ1503の第二区分1505が換字・転置されて、128ビットのデータに変換される。その後、 π 関数処理部1519において、ワーク鍵1525の一部をパラメータとして、 π 関数処理部1518で生成された128ビットのデータが換字・転置されて、128ビットのデータに変換される。このデータが、マスキングデータ1520の3番目のデータ g_2 1523となる。また、 π 関数処理部1518で生成された128ビットのデータが、伸長データの3番目の128ビットデータである第三区分（図示せず）を暗号化するためのワーク鍵1526となる。

【0247】上記の処理を伸長データを構成する全ての区分（128ビットデータ）に対して行うことで、マスキングデータ1520を生成する。

【0248】ここで、図15に示すマスキング装置1501において、 π 関数処理部1513、1514、1518、1519、・・・は、図13に示す第二実施形態で用いた π 関数処理部1306～1313と同じものである。

【0249】したがって、本発明の第三実施形態によれば、換字・転置処理にDESなどの暗号化関数を用いた

10

20

30

40

50

場合に比べて、データ攪乱度の高いマスクデータを、迅速に生成することができる。

【0250】また、本発明の第三実施形態において、マスクデータ1520は、共通鍵1502を伸長させたデータと見ることができる。また、マスクデータ1520は、共通鍵1502によって伸長データ1503を暗号化したデータと見ることができる。すなわち、図15に示す処理と逆の処理を行うことで、共通鍵1502およびマスクデータ1520から、伸長データ1503を復号することができる。

【0251】なお、本発明の第三実施形態では、共通鍵1502の長さを128ビットとしたが、本発明はこれに限定されない。また、伸長データ1503の各区分1504、1505、・・・の各々に対して、 π 関数処理部での処理を2回行うようにしたが、本発明はこれに限定されるものではない。

【0252】次に、本発明の第四実施形態について説明する。

【0253】本実施形態は、電子メールなどのデータ暗号化システムに関するものであり、データ暗号化装置と、データ復号化装置とを含む。なお、以下に説明するデータ暗号化装置およびデータ復号化装置は、第一実施形態のハッシュ値生成装置と同様に、パーソナルコンピュータやICカードなどのマイクロプロセッサを備えた情報処理装置において、マイクロプロセッサに所定のプログラムを実行させることで実現可能である。また、1つのLSIで実現させることも可能である。

【0254】まず、データ暗号化装置について説明する。

【0255】図16は、本発明の第四実施形態であるデータ暗号化システムを構成するデータ暗号化装置の機能構成を示す図である。

【0256】図16に示すデータ暗号化装置1601には、楕円曲線暗号におけるパラメータであるベースポイントP1602、公開鍵Q1603、および、平文1604が入力される。

【0257】ここで、楕円曲線暗号とは、次式、 $y^2 = x^3 + ax + b$

で表される楕円曲線上の2点 (x_1, y_1) 、 (x_2, y_2) の加算 $(x_1, y_1) + (x_2, y_2)$ や、整数倍演算 $k(x_1, y_1)$ などを定義することにより生成される公開鍵暗号のことである。

【0258】ベースポイントP1602や公開鍵Q1603も、上記の楕円曲線上の点であり、後で述べる秘密鍵d1802との間で次の関係を満たす。

【0259】 $Q = dP$

データ暗号化装置1601に入力されたベースポイントP1602は、乱数生成装置1607で生成された乱数kとともに、整数倍演算部1608へ入力される。これを受けて、整数倍演算部1608は、次式で示される

処理を実行することで、データR1617を生成する。

【0260】 $R = kP$

このデータR1617は、暗号文1616の最初のデータとなる。

【0261】また、データ暗号化装置1601に入力された公開鍵Q1603は、乱数生成装置1607で生成された乱数kとともに、整数倍演算部1609へ入力される。これを受けて、整数倍演算部1609は、次式で示される処理を実行することで、上記の楕円曲線上の点 (x, y) を生成する。

【0262】 $(x, y) = kQ$

また、データ暗号化装置1601に入力された平文1604のうち、最初のNビットデータ1605は、圧縮・暗号化部1612に入力される。これを受けて、圧縮・暗号化部1612は、ハッシュ値生成装置1611で生成されたハッシュ値を鍵として、最初のNビットデータ1605の圧縮・暗号化処理を行う。これによりデータC11618を生成する。このデータC11618は、暗号文1616の2番目のデータとなる。

【0263】なお、ハッシュ値生成装置1611は、番号生成部1610で生成されたシーケンス番号「1」と、整数倍演算部1609で生成した (x, y) のうちの数値xとを接続することで得られるデータに対するハッシュ値を生成する。

【0264】また、データ暗号化装置1601に入力された平文1604のうち、2番目のNビットデータ1606は、圧縮・暗号化部1615に入力される。これを受けて、圧縮・暗号化部1615は、ハッシュ値生成装置1614で生成されたハッシュ値を鍵として、2番目のNビットデータ1606の圧縮・暗号化処理を行う。これによりデータC21619を生成する。このデータC21619は、暗号文1616の3番目のデータとなる。

【0265】なお、ハッシュ値生成装置1614は、番号生成部1613で生成されたシーケンス番号「2」と、整数倍演算部1609で生成した (x, y) のうちの数値xとを接続することで得られるデータに対するハッシュ値を生成する。

【0266】上記の処理を平文1604を構成する全てのNビットデータに対して行うことで、暗号文1616を生成する。

【0267】なお、図16に示すデータ暗号化装置1601において、ハッシュ値生成装置1611、1614、・・・は、本発明の第一実施形態で説明したものでもよいし、従来の「ブロック暗号を利用したハッシュ関数」や「専用ブロック関数」であってもよい。

【0268】次に、図16に示す圧縮・暗号化部1612、1615、・・・での処理について説明する。

【0269】図17は、図16に示す圧縮・暗号化部1612、1615、・・・の機能構成を示す図である。

10

20

30

40

50

【0270】ここで、Nビットデータとは、図16において、平文1604を構成するNビットデータ1605、1606、・・・に相当する。また、データCは、図16において、圧縮・暗号化部1612、1615、・・・が生成するデータC₁1618、C₂1619、・・・を示す。さらに、鍵1705とは、図16において、対応するハッシュ値生成装置1611、1614、・・・が生成したハッシュ値に相当する。

【0271】図17において、鍵1705が入力されると、拡張部1706はこの鍵1705を受け取ってコピーを複数個生成し、これ等を繋ぎ合わせてワーク鍵1723を生成する。

【0272】また、Nビットデータの最初のデータ区分である第1区分1703は、圧縮処理部1707において、ワーク鍵1723の一部をパラメータとして、ハフマン圧縮などにより圧縮（換字）処理が施される。そして、その結果が、128ビットの圧縮データ1708および端数データ1706として出力される。

【0273】128ビットの圧縮データ1708は、 π 関数処理部1710において、ワーク鍵1723の一部をパラメータとして、換字・転置処理が施され、128ビットのデータに変換される。その後、 π 関数処理部1711において、ワーク鍵1723の一部をパラメータとして、さらに換字・転置処理が施されて、128ビットのデータに変換される。このデータが、生成すべきデータCの最初のデータb₁1720となる。また、 π 関数処理部1710で生成された128ビットのデータは、拡張部1712に入力され、複数コピーされる。そして、これ等が繋ぎ合わされて、Nビットデータの2番目のデータ区分である第2区分1704を暗号化するためのワーク鍵1722となる。

【0274】また、Nビットデータの2番目のデータ区分である第2区分1704は、圧縮処理部1713において、ワーク鍵1722の一部をパラメータとして、ハフマン圧縮などにより圧縮（換字）処理が施される。そして、その結果が、圧縮データ1714および端数データ1715として出力される。ここで、圧縮データ1714は、第1区分1703を圧縮（換字）処理した際に生成された端数データ1709との合計ビット長が、128ビットとなるように生成される。

【0275】圧縮データ1715は、第1区分1703を圧縮（換字）処理した際に生成された端数データ1709とを繋ぎ合わされて、128ビットのデータになる。その後、 π 関数処理部1716において、ワーク鍵1722の一部をパラメータとして、換字・転置処理が施され、128ビットのデータに変換される。さらにその後、 π 関数処理部1718において、ワーク鍵1722の一部をパラメータとして、さらに換字・転置処理が施されて、128ビットのデータに変換される。このデータが、生成すべきデータCの2番目のデータb₂17

21となる。また、 π 関数処理部1716で生成された128ビットのデータは、拡張部1717に入力され、複数コピーされる。そして、これ等が繋ぎ合わされて、Nビットデータの3番目のデータ区分を暗号化するためのワーク鍵となる。

【0276】上記の処理をNビットデータを構成する全ての区分に対して行うことで、対応するデータCを生成する。

【0277】ここで、図17に示す π 関数処理部1710、1711、1716、1718、・・・は、図13に示す第二実施形態で用いた π 関数処理部1306、1307、1308、・・・1313と同じものである。

【0278】本発明の第四実施形態を構成するデータ暗号化装置では、楕円暗号とハッシュ値生成装置とを組み合わせることで、データの暗号化を行っている。くわえて、圧縮・暗号化部として、第二実施形態で用いた π 関数処理部を利用している。

【0279】したがって、本発明の第四実施形態を構成するデータ暗号化装置によれば、従来のRSA（Rivest, Shamir, Adleman）のような公開鍵暗号方式のデータ暗号化装置に比べて、長いデータに対し、データ攪乱度の高い暗号文を、迅速に生成することができる。

【0280】なお、このデータ暗号化装置は、公開鍵Q1603を用いて、平文1604を暗号文1616に変換するという点において、従来のRSAの公開鍵暗号方式を用いたデータ暗号化装置と共通する。しかしながら、本発明の第四実施形態を構成するデータ暗号化装置では、上述したように、 π 関数処理部を用いて換字・転置処理を行っている点で従来のものと異なる。

【0281】また、圧縮処理部1707、1713、・・・において、ハフマン圧縮などにより圧縮（換字）処理を行っているので、このハフマン圧縮などにより圧縮することができる通常の平文に対しては、暗号文1616の長さが平文1604の長さよりも短くなる。この点でも、従来のものと異なる。

【0282】ところで、図17に示す圧縮・暗号化部1612、1615、・・・では、Nビットデータを構成する各区分（第1区分1703、第2区分1704、・・・）に対して、 π 関数処理部での変換処理を2回行うようにしている。しかしながら、本発明はこれに限定されるものではない。

【0283】次に、データ復号化装置について説明する。

【0284】図18は、本発明の第四実施形態であるデータ暗号化システムを構成するデータ復号化装置の機能構成を示す図である。

【0285】図18に示すデータ復号化装置1801には、楕円曲線暗号におけるパラメータである秘密鍵d1802、および、暗号文1803が入力される。

【0286】データ復号化装置1801に入力された秘密鍵d1802は、同じくデータ復号化装置1801に入力された暗号文1803のうちの最初のデータR1804（これは、図16におけるデータR1617に相当）とともに、整数倍演算部1807へ入力される。これを受けて、整数倍演算部1807は、次式で示される処理を実行することで、楕円曲線上の点(x, y)を生成する。

$$【0287】(x, y) = kR$$

また、データ復号化装置1801に入力された暗号文1803のうち、2番目のデータC₁1805は、復号・伸長化部1810に入力される。これを受けて、復号・伸長化部1810は、ハッシュ値生成装置1809で生成されたハッシュ値を鍵として、2番目のデータC₁1805の復号・伸長化処理を行う。これにより、Nビットのデータを生成する。このデータは、平文1814の最初のNビットデータ1815となる。

【0288】なお、ハッシュ値生成装置1809は、番号生成部1808で生成されたシーケンス番号「1」と、整数倍演算部1807で生成した(x, y)のうちの数値xとを接続することで得られるデータに対するハッシュ値を生成する。

【0289】また、データ復号化装置1801に入力された暗号文1803のうち、3番目のデータC₂1806は、復号・伸長化部1813に入力される。これを受けて、復号・伸長化部1813は、ハッシュ値生成装置1812で生成されたハッシュ値を鍵として、3番目のデータC₂1806の復号・伸長化処理を行う。これにより、Nビットのデータを生成する。このデータは、平文1814の2番目のNビットデータ1816となる。

【0290】なお、ハッシュ値生成装置1812は、番号生成部1811で生成されたシーケンス番号「2」と、整数倍演算部1807で生成した(x, y)のうちの数値xとを接続することで得られるデータに対するハッシュ値を生成する。

【0291】上記の処理を暗号文1803を構成する2番目のデータC₁1805から最後のデータまでに対して行うことで、平文1814を生成する。

【0292】なお、図18に示すデータ復号化装置1801において、ハッシュ値生成装置1809、1812、・・・は、図16に示すものと同じである。

【0293】次に、図18に示す復号・伸長化部1810、1813、・・・での処理について説明する。

【0294】図19は、図18に示す復号・伸長化部1810、1813、・・・の機能構成を示す図である。

【0295】ここで、データCは、図18において、暗号文1803の2番目以降のデータC₁1805、C₂1806、・・・を示す。また、Nビットデータとは、図18において、復号・伸長化部1810、1813、・・・が生成したNビットデータ1815、1816、

・・・に相当する。さらに、鍵1905とは、図18において、対応するハッシュ値生成装置1809、1812、・・・が生成したハッシュ値に相当する。

【0296】また、 π^{-1} 関数処理部1907、1910、1914、1916、・・・は、図17に示す π 関数処理部1710、1711、1716、1718、・・・での処理と逆関数の関係にある処理を行う。すなわち、 π^{-1} 関数は、 π 関数の逆関数である。

【0297】同じパラメータを π 関数と π^{-1} 関数に設定した場合、データmを π 関数によって変換して得られるデータ $\pi(m)$ を、さらに、 π^{-1} 関数によって変換すると、元のデータmに戻る。すなわち、 π^{-1} 関数は π 関数と次式のような関係にある。

$$【0298】m = \pi^{-1}(\pi(m))$$

また、伸長処理部1911、1917、・・・は、図17に示す圧縮処理部1707、1713、・・・での処理と逆変換の関係にある処理を行う。

【0299】同じパラメータを圧縮処理部と伸長処理部に設定した場合、データmを圧縮（換字）処理によって変換して得られるデータを、さらに、伸長（換字）処理によって変換すると、元のデータmに戻る。

【0300】図19において、鍵1905が入力されると、拡張部1906はこの鍵1905を受け取ってコピーを複数個生成し、これ等を繋ぎ合わせてワーク鍵1923を生成する。

【0301】また、データCの最初の128ビットデータb₁1903は、 π^{-1} 関数処理部1907において、ワーク鍵1923の一部をパラメータとして、換字・転置処理が施されて128ビットデータ変換された後、 π^{-1} 関数処理部1910において、ワーク鍵1923の一部をパラメータとして、さらに換字・転置処理が施されて、128ビットデータに変換される。

【0302】 π^{-1} 関数処理部1910の出力結果は、伸長処理部1911において、ワーク鍵1923の一部をパラメータとして、伸長（換字）処理が施される。そして、その結果が、128ビットの伸長データ1912およびその端数データ1913として出力される。この128ビットの伸長データが、生成すべきNビットデータのうちの第1区分のデータ1921となる。また、 π^{-1} 関数処理部1907で生成された128ビットのデータは、拡張部1909に入力され、複数コピーされる。そして、これ等が繋ぎ合わされて、データCの2番目の128ビットデータb₂1904を復号化するためのワーク鍵1924となる。

【0303】また、データCの2番目の128ビットデータb₂1904は、 π^{-1} 関数処理部1914において、ワーク鍵1924の一部をパラメータとして、換字・転置処理が施されて128ビットデータに変換された後、 π^{-1} 関数処理部1916において、ワーク鍵1924の一部をパラメータとして、さらに換字・転置処理が

施されて、128ビットデータに変換される。

【0304】 π^{-1} 関数処理部1916の出力結果は、伸長処理部1917において、ワーク鍵1924の一部をパラメータとして、伸長（換字）処理が施される。そして、その結果が、伸長データ1918およびその端数データ1919として出力される。

【0305】ここで、伸長データ1918は、データb11903を伸長（換字）処理した際に生成された端数データ1913との合計ビット長が、128ビットとなるように生成される。この伸長データ1918と端数データ1913とを繋いだ合計128ビットのデータが、生成すべきNビットデータのうちの第2区分のデータ1922となる。また、 π^{-1} 関数処理部1914で生成された128ビットのデータは、拡張部1915に入力され、複数コピーされる。そして、これ等が繋ぎ合わされて、データCの3番目の128ビットデータを復号化するためのワーク鍵となる。

【0306】上記の処理をデータCを構成する全ての128ビットデータb1、b2、・・・に対して行うことでNビットデータを生成する。

【0307】本発明の第四実施形態を構成するデータ復号化装置は、秘密鍵d1802を用いて、暗号文1803を平文1814に復号変換するという点において、従来のRSAの公開鍵暗号方式を用いたデータ復号化装置と共通する。しかしながら、本発明の第四実施形態を構成するデータ暗号化装置では、上述したように、 π^{-1} 関数処理部を用いて換字・転置処理を行っている点で従来のものと異なる。

【0308】また、伸長処理部1911、1917、・・・において、伸長（換字）処理を行っているので、このハフマン圧縮などで圧縮された暗号文に対して、復号化された平文1814は、そのデータ長が長くなる。この点でも、従来のものと異なる。

【0309】次に、上記の第四実施形態におけるデータ暗号化装置の変形例について説明する。

【0310】図20は、図16に示す本発明の第四実施形態におけるデータ暗号化装置の変形例の機能構成を示す図である。

【0311】図20に示すデータ暗号化装置2001には、楕円曲線暗号におけるパラメータであるベースポイントP2002、公開鍵Q2003、および、平文2004が入力される。

【0312】データ暗号化装置2001に入力されたベースポイントP2002は、乱数生成装置2007で生成された乱数kとともに、整数倍演算部2008へ入力される。これを受けて、整数倍演算部2008は、次式で示される処理を実行することで、データR2017を生成する。

$$【0313】R = kP$$

このデータR2017は、暗号文2016の最初のデー

タとなる。

【0314】また、データ暗号化装置2001に入力された公開鍵Q2003は、乱数生成装置2007で生成された乱数kとともに、整数倍演算部2009へ入力される。これを受けて、整数倍演算部2009は、次式で示される処理を実行することで、楕円曲線上の点(x, y)を生成する。

$$【0315】(x, y) = kQ$$

また、データ暗号化装置2001に入力された平文2004のうち、最初のNビットデータ2005は、圧縮・暗号化部2012に入力される。これを受けて、圧縮・暗号化部2012は、ハッシュ値生成装置2011で生成されたハッシュ値を鍵2020として、最初のNビットデータ2005の圧縮・暗号化処理を行う。これによりデータC12018を生成する。このデータC12018は、暗号文2016の2番目のデータとなる。

【0316】なお、ハッシュ値生成装置2011は、整数倍演算部2009で生成した(x, y)のうちの数値xに対するハッシュ値を生成する。このハッシュ値が上記の鍵2020となる。

【0317】また、データ暗号化装置2001に入力された平文2004のうち、2番目のNビットデータ2006は、圧縮・暗号化部2015に入力される。これを受けて、圧縮・暗号化部2015は、ハッシュ値生成装置2014で生成されたハッシュ値を鍵2021として、2番目のNビットデータ2006の圧縮・暗号化処理を行う。これによりデータC22019を生成する。このデータC22019は、暗号文2016の3番目のデータとなる。

【0318】なお、ハッシュ値生成装置2014は、鍵2020に対するハッシュ値を生成する。このハッシュ値が上記の鍵2021となる。

【0319】上記の処理を平文2004を構成する全てのNビットデータに対して行うことで、暗号文2016を生成する。

【0320】なお、図16に示すデータ暗号化装置1601において、圧縮・暗号化部2012、2015、・・・には、図16に示す圧縮・暗号化部1612、1615、・・・と同じものが用いられる。

【0321】次に、本発明の第五実施形態について説明する。

【0322】本実施形態は、本発明の第四実施形態と同様、電子メールなどのデータ暗号化システムに関するものであり、データ暗号化装置と、データ復号化装置とを含む。なお、以下に説明するデータ暗号化装置およびデータ復号化装置は、第一実施形態のハッシュ値生成装置と同様に、パーソナルコンピュータやICカードなどのマイクロプロセッサを備えた情報処理装置において、マイクロプロセッサに所定のプログラムを実行させることで実現可能である。また、1つのLSIで実現させるこ

とも可能である。

【0323】まず、データ暗号化装置について説明する。

【0324】図21は、本発明の第五実施形態であるデータ暗号化システムを構成するデータ暗号化装置の機能構成を示す図である。

【0325】図21に示すデータ暗号化装置2101には、楕円曲線暗号におけるパラメータであるベースポイントP2102、公開鍵Q12103、公開鍵Q22104、公開鍵Q32105、公開鍵Q42123、および、平文2106が入力される。

【0326】データ暗号化装置2101に入力されたベースポイントP2102は、乱数生成装置2113で生成された乱数kとともに、整数倍演算部2123へ入力される。これを受けて、整数倍演算部2123は、次式で示される処理を実行することで、データR2109を生成する。

【0327】 $R = kP$ このデータR2109は、暗号文2108の最初のデータとなる。

【0328】また、データ暗号化装置2101に入力された公開鍵Q12103は、乱数生成装置2113で生成された乱数kとともに、整数倍演算部2114へ入力される。これを受けて、整数倍演算部2114は、次式で示される処理を実行することで、上記の楕円曲線上の点 (x_1, y_1) を生成する。

【0329】 $(x_1, y_1) = kQ_1$

この (x_1, y_1) のうち、数値 x_1 は、その後、ハッシュ値生成装置2119に入力されて、ハッシュ値 $h(x_1)$ に変換される。

【0330】同様に、データ暗号化装置2101に入力された公開鍵Q22104は、乱数生成装置2113で生成された乱数kとともに、整数倍演算部2115へ入力される。これを受けて、整数倍演算部2115は、次式で示される処理を実行することで、上記の楕円曲線上の点 (x_2, y_2) を生成する。

【0331】 $(x_2, y_2) = kQ_2$

この (x_2, y_2) のうち、数値 x_2 は、その後、ハッシュ値生成装置2126に入力されて、ハッシュ値 $h(x_2)$ に変換される。

【0332】また、同様に、データ暗号化装置2101に入力された公開鍵Q32105は、乱数生成装置2113で生成された乱数kとともに、整数倍演算部2116へ入力される。これを受けて、整数倍演算部2116は、次式で示される処理を実行することで、上記の楕円曲線上の点 (x_3, y_3) を生成する。

【0333】 $(x_3, y_3) = kQ_3$

この (x_3, y_3) のうち、数値 x_3 は、その後、ハッシュ値生成装置2127に入力されて、ハッシュ値 $h(x_3)$ に変換される。

【0334】さらに、同様に、データ暗号化装置210

1に入力された公開鍵Q42123は、乱数生成装置2113で生成された乱数kとともに、整数倍演算部2124へ入力される。これを受けて、整数倍演算部2124は、次式で示される処理を実行することで、上記の楕円曲線上の点 (x_4, y_4) を生成する。

【0335】 $(x_4, y_4) = kQ_4$

この (x_4, y_4) のうち、数値 x_4 は、その後、ハッシュ値生成装置2128に入力されて、ハッシュ値 $h(x_4)$ に変換される。

【0336】しきい値ロジック部2125は、ハッシュ値生成装置2126、2127、2128で生成したハッシュ値 $h(x_2)$ 、 $h(x_3)$ 、 $h(x_4)$ のうち、いずれか2つのハッシュ値が分かれば、ハッシュ値生成装置2119で生成したハッシュ値 $h(x_1)$ を求めることができる条件式を満たす値 f_12110 、 f_22111 を生成する。この値 f_12110 、 f_22111 は、それぞれ、暗号文2108の2番目、3番目のデータとなる。

【0337】なお、このしきい値ロジック部2125の詳細については後述する。

【0338】データ暗号化装置2101に入力された平文2106のうち、最初のNビットデータ2107は、圧縮・暗号化部2120に入力される。これを受けて、圧縮・暗号化部2120は、ハッシュ値生成装置2119で生成されたハッシュ値 $h(x_1)$ を鍵2120として、最初のNビットデータ2107の圧縮・暗号化処理（これについては後述する）を行う。これによりデータC12112を生成する。このデータC12112は、暗号文2108の4番目のデータとなる。また、鍵2120は、ハッシュ値生成装置2121に入力されて、平文2106の2番目のNビットデータを暗号化するための鍵2122に変換される。

【0339】上記の処理を平文2106を構成する全てのNビットデータに対して行うことで、暗号文210812を生成する。

【0340】なお、図21に示すデータ暗号化装置2101において、ハッシュ値生成装置は、本発明の第一実施形態で説明したものでもよいし、従来の「ブロック暗号を利用したハッシュ関数」や「専用ブロック関数」であってもよい。

【0341】次に、図21に示すしきい値ロジック部2125での処理について説明する。

【0342】上述したように、しきい値ロジック部2125は、ハッシュ値 $h(x_2)$ 、 $h(x_3)$ 、 $h(x_4)$ のうちのいずれか2つのハッシュ値が分かれば、ハッシュ値 $h(x_1)$ を求めることができる条件式を満たす値 f_1 、 f_2 を生成する。

【0343】図22は、図21に示すしきい値ロジック部2125の機能構成を示す図である。

【0344】図22に示すように、しきい値ロジック部

10

20

30

40

50

2125には、5つのデータ $q_{1x}2202$ 、 $h(x_1)2203$ 、 $h(x_2)2204$ 、 $h(x_3)2205$ 、 $h(x_4)2206$ が入力される。

【0345】ここで、 $q_{1x}2202$ は、図21において、公開鍵 $Q12103$ の x 座標値である。また、 $h(x_1)2203$ 、 $h(x_2)2204$ 、 $h(x_3)2205$ 、 $h(x_4)2206$ は、それぞれ、ハッシュ値生成装置2119、2126、2127、2128で生成されたハッシュ値である。

【0346】演算部2208は、次式で示される処理を実行することで、データ f_12110 を生成する。

【0347】 $f_1 = g(q_{1x}, h(x_1), h(x_2), h(x_3), h(x_4))$

また、演算部2210は、次式で示される処理を実行することで、データ f_22111 を生成する。

【0348】 $f_2 = g(h(q_{1x}), h(x_1), h(x_2), h(x_3), h(x_4))$

なお、上記の式 f_1 、 f_2 において、関数 g は、 $g(x, a_1, a_2, a_3, a_4) = a_1 + a_2 \cdot x + a_3 \cdot x^2 + a_4 \cdot x^3 \pmod{n}$ で定義されるものである。

【0349】したがって、演算部2208、2210で生成されたデータ f_1 、 f_2 は、 q_{1x} 、および、 $h(x_1)$ 、 $h(x_2)$ 、 $h(x_3)$ 、 $h(x_4)$ との間において、以下に示す四元連立方程式が成立する。

【0350】 $f_1 = h(x_1) + h(x_2) \cdot q_{1x} + h(x_3) \cdot q_{1x}^2 + h(x_4) \cdot q_{1x}^3 \pmod{n}$
 $f_2 = h(x_1) + h(x_2) \cdot h(q_{1x}) + h(x_3) \cdot h(q_{1x})^2 + h(x_4) \cdot h(q_{1x})^3 \pmod{n}$

よって、 f_1 、 f_2 、 q_{1x} の値が既知である場合、 $h(x_1)$ 、 $h(x_2)$ 、 $h(x_3)$ 、 $h(x_4)$

のうちのいずれか二つの値が分かれば、他の二つの値も分かることになる（未知数2、方程式数2の連立方程式となるため）。

【0351】次に、図21に示す圧縮・暗号化部2120、・・・での処理について説明する。

【0352】図23は、図21に示す圧縮・暗号化部2120、・・・の機能構成を示す図である。

【0353】ここで、 N ビットデータとは、図21において、平文2106を構成する N ビットデータ2107、・・・に相当する。また、データ C は、図21において、圧縮・暗号化部2120、・・・が生成するデータ C_12112 、・・・を示す。さらに、鍵2309とは、図21において、鍵2120、2122、・・・に相当する。

【0354】図23において、鍵2309が入力されると、拡張部2310はこの鍵2309を受け取ってコピーを複数個生成し、これ等を繋ぎ合わせてワーク鍵2311を生成する。

【0355】また、 N ビットデータの最初のデータ区分である第1区分2303は、圧縮処理部2312にお

て、ワーク鍵2311の一部をパラメータとして、ハフマン圧縮などにより圧縮（換字）処理が施される。そして、その結果が、128ビットの圧縮データ2313およびその端数データ2314として出力される。

【0356】128ビットの圧縮データ2313は、 π 関数処理部2315において、ワーク鍵2311の一部をパラメータとして、ブロック暗号処理が施され、128ビットのデータに変換される。その後、 π 関数処理部2316において、ワーク鍵2311の一部をパラメータとして、さらにブロック暗号処理が施されて、128ビットのデータに変換される。このデータが、生成すべきデータ C の最初のデータ b_12306 となる。また、 π 関数処理部2315で生成された128ビットのデータは、拡張部2317に入力され、複数コピーされる。そして、これ等が繋ぎ合わされて、 N ビットデータの2番目のデータ区分である第2区分2304を暗号化するためのワーク鍵2318となる。

【0357】また、 N ビットデータの2番目のデータ区分である第2区分2304は、圧縮処理部2319において、ワーク鍵2318の一部をパラメータとして、ハフマン圧縮などにより圧縮（換字）処理が施される。そして、その結果が、圧縮データ2320および端数データ2321として出力される。ここで、圧縮データ2320は、第1区分2303を圧縮（換字）処理した際に生成された端数データ2314との合計ビット長が、128ビットとなるように生成される。

【0358】圧縮データ2320は、第1区分2303を圧縮（換字）処理した際に生成された端数データ2314とを繋ぎ合わされて、128ビットのデータになる。その後、 π 関数処理部2322において、ワーク鍵2318の一部をパラメータとして、ブロック暗号処理が施され、128ビットのデータに変換される。その後、 π 関数処理部2324において、ワーク鍵2318の一部をパラメータとして、さらにブロック暗号処理が施されて、128ビットのデータに変換される。このデータが、生成すべきデータ C の2番目のデータ b_22307 となる。

【0359】なお、第2区分2304を圧縮（換字）処理した際に生成された端数データ2321が r ビット（ $r \geq 1$ ）である場合、ハッシュ値生成装置2323は、鍵2309に対するハッシュ値を生成する。これを受けて、演算部2325は、ハッシュ値生成装置2323で生成されたハッシュ値の上位 r ビットと、端数データ2321との排他的論理的をと、 r ビットのデータを生成する。このデータが、生成すべきデータ C の3番目のデータ b_32308 となる。

【0360】上記の処理により、 N ビットデータに対する暗号データ C が生成される。

【0361】本発明の第五実施形態を構成するデータ暗号化装置では、平文2106の暗号化には、初期値とし

10

20

30

40

50

て、ハッシュ値 $h(x_1)$ を用いている。したがって、このデータ暗号化装置で暗号化された暗号文2108を復号するためには、ハッシュ値 $h(x_1)$ を求めればよい。

【0362】ところで、このデータ暗号化装置では、しきい値はブロック部2125において、ハッシュ値 $h(x_1)$ 、 $h(x_2)$ 、 $h(x_3)$ 、 $h(x_4)$ のうちのいずれか2つのハッシュ値が分かれば、ハッシュ値 $h(x_1)$ を求めることができる条件式を満たす値 f_1 、 f_2 を生成し、この f_1 、 f_2 を暗号文2108に付加している。

【0363】このため、暗号文2108を受信した者（すなわち、 f_1 、 f_2 を取得した者）は、 $h(x_2)$ 、 $h(x_3)$ および $h(x_4)$ のうちのいずれか2つ（ただし、図22に示す例でいえば、さらに q_{1x} が必要）が分かれば、暗号化に用いた $h(x_1)$ を求めることができる。

【0364】よって、暗号文2108を復号は、公開鍵 Q_1 と対になる秘密鍵 d_1 を所有する（単独でハッシュ値 $h(x_1)$ を求めることができる）者のみならず、公開鍵 Q_2 と対になる秘密鍵 d_2 を所有する（単独でハッシュ値 $h(x_2)$ を求めることができる）者、公開鍵 Q_3 と対になる秘密鍵 d_3 を所有する（単独でハッシュ値 $h(x_3)$ を求めることができる）者、および、公開鍵 Q_4 と対になる秘密鍵 d_4 を所有する（単独でハッシュ値 $h(x_4)$ を求めることができる）者のうちの、いずれか二人が協力することでも行うことができる。

【0365】前者は、1 out of 1 の復号ロジックであり、後者は2 out of 3 の復号ロジックである。本実施形態によれば、このように、復号可能となる受信者数のしきい値制御が可能となる。

【0366】すなわち、暗号文2108を秘密鍵 d_1 、 d_2 、 d_3 、 d_4 の所有者各々に対して同報通信（マルチキャスト）することで、秘密鍵 d_1 の持ち主が単独で復号できるとともに、秘密鍵 d_2 、 d_3 、 d_4 の持ち主3人のうちの2人が協力すれば復号可能なようにすることができる。

【0367】次に、図21に示すデータ暗号化装置の変形例について説明する。

【0368】図25は図21に示すデータ暗号化装置の変形例の機能構成を示す図である。

【0369】ここで、図21に示すデータ暗号化装置2101と同じ機能を有するものには、同じ符号を付している。

【0370】図25に示すデータ暗号化装置2101aは、整数倍演算部2123に代えて整数倍演算部2501を用いた点で、図21に示すデータ暗号化装置2101と異なる。その他の構成は、図21に示すものと同じである。

【0371】整数倍演算部2501は、ベースポイント P_{2102} と、乱数生成装置2113で生成された乱数

k とを受け取って、 $R = kP$ で示される処理を実行する点で、図21に示す整数倍演算部2123と同じである。しかしながら、上記の式で求められる R のうち、 x 座標の値 R_x 2502のみを出力する点で異なる。

【0372】したがって、図25に示すデータ暗号化装置2101aでは、このデータ R_x 2502が、暗号文2108aの最初のデータとなる。

【0373】図25に示すデータ暗号化装置2101aでは、図21に示すデータ暗号化装置2101に比べて、生成される暗号文の長さが幾分短くなる。

【0374】すなわち、図21に示すデータ暗号化装置2101では、整数倍演算部2123において、楕円曲線上の点 $R = kP$ を求め、求めた R の x 座標値 R_x および y 座標値 R_y を出力して、暗号文2108の最初のデータ R_{2109} としていた。

【0375】これに対し、図25に示すデータ暗号化装置2101aでは、整数倍演算部2501において、楕円曲線上の点 $R = kP$ を求め、求めた R のうちの x 座標値 R_x のみを出力して、暗号文2108aの最初のデータ R_x 2502としている。

【0376】したがって、図25に示すデータ暗号化装置2101aで生成された暗号文2108aは、図21に示すデータ暗号化装置2101で生成された暗号文2108よりも、 y 座標値 R_y のデータ分（たとえば、160ビット）だけ短くなる。

【0377】次に、データ復号化装置について説明する。

【0378】図26は、本発明の第五実施形態であるデータ暗号化システムを構成するデータ復号化装置の機能構成を示す図である。このデータ復号化装置は、図25に示すデータ暗号化装置2101aで生成された暗号文2108aを復号するためのものである。

【0379】データ復号化装置は、図26に示すように、データ回復機関2607と、鍵保管機関A2608と、鍵保管機関B2609とで構成される。

【0380】図26に示すデータ回復機関2607に、公開鍵 Q_{12103} と、暗号文2108aとが入力されると、演算部2611は、暗号文2108aの最初のデータ R_x 2502との間で、次の楕円曲線方程式を満足する y 座標値 R_y を求める。

【0381】 $R_y^2 = R_x^3 + a \cdot R_x + b$ 通常、この方程式の解 R_y は2つ存在する。一方の解を r とすると、他方は $-r$ となる（ただし、楕円曲線として $y^2 + xy = x^3 + ax + b$ を用いる場合には、一方の解を r とすると、他方は $R_x + r$ となる）。

【0382】どちらか任意の一方をとり、他方を無視する。ここでは、 r の方をとるものとする。演算部2611は、 $R = (R_x, r)$ を出力する。

【0383】次に、データ回復機関2607は、演算部2611から出力された値 R を鍵保管機関A2608、

B2609に各々送信する。

【0384】これを受けて、鍵保管機関A2608、以下に示す処理を順次行う。

【0385】自己が格納している秘密鍵 d_22618 （公開鍵 Q_22104 と対）を読み出す。

【0386】演算部2619において、データ回復機関2607から送られてきた値 R との間で、次式を満たす楕円曲線上の点 (x_2, y_2) を求める。

【0387】 $(x_2, y_2) = d_2R$

ハッシュ値生成装置2620において、演算部2619で求めた (x_2, y_2) のうちの x 座標値 x_2 のハッシュ値 $h(x_2)$ を生成する。

【0388】ハッシュ値生成装置2620で生成したハッシュ値 $h(x_2)$ を、データ回復機関2607へ送信する。

【0389】また、鍵保管機関B2609は、以下に示す処理を順次行う。

【0390】自己が格納している秘密鍵 d_32621 （公開鍵 Q_32105 と対）を読み出す。

【0391】演算部2622において、データ回復機関2607から送られてきた値 R との間で、次式を満たす楕円曲線上の点 (x_3, y_3) を求める。

【0392】 $(x_3, y_3) = d_3R$

ハッシュ値生成装置2623において、演算部2622で求めた (x_3, y_3) のうちの x 座標値 x_3 のハッシュ値 $h(x_3)$ を生成する。

【0393】ハッシュ値生成装置2623で生成したハッシュ値 $h(x_3)$ を、データ回復機関2607へ送信する。

【0394】上記のような処理の結果、得られたハッシュ値 $h(x_2)$ 、 $h(x_3)$ は以下に示すような特徴を有する。

【0395】かりに、鍵保管機関A2608、B2609に各々送信されるデータが、 $R = (R_x, r)$ ではなく、 $R' = (R_x, -r)$ であったとする。

【0396】この場合、鍵保管機関A2608の演算部2919では、 $(x_2', y_2') = d_2R'$ の計算が行われことになる。

【0397】しかし、楕円曲線上の演算の性質より、 $(x_2', y_2') = (x_2, -y_2)$ となる。

【0398】すなわち、楕円曲線 $y^2 = x^3 + ax + b$ を用いた演算では、 $-(x, y) = (x, -y)$ が成立する。このとき、 $(x_2, y_2) = d_2(R_x, r)$ であったとすると、 $d_2(R_x, -r) = d_2(-(R_x, r)) = -d_2(R_x, r) = (x_2, -y_2)$ となる。

【0399】また、楕円曲線 $y^2 + xy = x^3 + ax + b$ を用いた演算では、 $-(x, y) = (x, x + y)$ が成立する。このとき、 $(x_2, y_2) = d_2(R_x, r)$ であったとすると、 $d_2(R_x, R_x + r) = d_2(-(R_x, r)) = -d_2(R_x, r) = (x_2, x_2 + y_2)$ となる。

【0400】したがって、 $h(x_2') = h(x_2)$ となる。

【0401】つまり、 $R = (R_x, r)$ を入力しても、あるいは $R' = (R_x, -r)$ を入力しても、鍵保管機関A2608が出力するハッシュ値 $h(x_2)$ は変化しない。

【0402】同様に、 $R = (R_x, r)$ を入力しても、あるいは $R' = (R_x, -r)$ を入力しても、鍵保管機関B2609が出力するハッシュ値 $h(x_3)$ も変化しない。

【0403】データ回復機関2607の演算部2611において、方程式の解 R_y が2つ存在するにもかかわらず、どちらか任意の一方のみを、鍵保管機関A2608および鍵保管機関B2609に送信したのは、このような理由による。

【0404】次に、データ回復機関2607は、鍵保管機関A2608および鍵保管機関B2609から各々ハッシュ値 $h(x_2)$ 、 $h(x_3)$ を受け取ると、しきい値逆算ロジック部2612による処理を開始する。

【0405】しきい値逆算ロジック部2612は、まず、ハッシュ値 $h(x_2)$ 、 $h(x_3)$ と、公開鍵 Q_12103 の x 座標値 q_{1x} と、暗号文2108aの2番目および3番目のデータ f_12110 、 f_22111 と、を受け取る。

【0406】そして、以下に示す2元連立方程式を満たす $h(x_1)$ 、 $h(x_4)$ を生成する。

【0407】 $f_1 = h(x_1) + h(x_2) \cdot q_{1x} + h(x_3) \cdot q_{1x}^2 + h(x_4) \cdot q_{1x}^3 \pmod{n}$

$f_2 = h(x_1) + h(x_2) \cdot h(q_{1x}) + h(x_3) \cdot h(q_{1x})^2 + h(x_4) \cdot h(q_{1x})^3 \pmod{n}$

ここで、上記の2元連立方程式は、図21および図25に示すハッシュ値生成装置で生成されたハッシュ値 $h(x_1)$ 、 $h(x_2)$ 、 $h(x_3)$ 、 $h(x_4)$ の値を未知とする4元連立方程式のうち、 $h(x_2)$ と $h(x_3)$ とを既知とした場合に相当する。

【0408】図21および図25に示すハッシュ値生成装置では、乱数生成装置で生成した乱数 k を用いて $(x_2, y_2) = kQ_2$ を求め、このうちの x 座標の数値 x_2 に対するハッシュ値を $h(x_2)$ としている。同様に、乱数 k を用いて $(x_3, y_3) = kQ_3$ を求め、このうちの x 座標の数値 x_3 に対するハッシュ値を $h(x_3)$ としている。

【0409】これに対し、図26に示すハッシュ値生成装置では、上記と同じ乱数 k を用いて $R = kP$ により求められた R （あるいは、この R と y 座標の符号だけ異なる R' ）を用いて、 d_2R （あるいは、 d_2R' ）を求め、このうちの x 座標の数値 x_2 （あるいは、 x_2' ）に対するハッシュ値を $h(x_2)$ としている。同様に、 d_3R （あるいは、 d_3R' ）を求め、このうちの x 座標の数値 x_3 （あるいは、 x_3' ）に対するハッシュ値を h

(x_3)としている。

【0410】ここで、楕円曲線暗号における秘密鍵と公開鍵との関係から、 $Q_2 = d_2 P$ 、 $Q_3 = d_3 P$ が成立する。したがって、

$$d_2 R = d_2 k P = k d_2 P = k Q_2 = (x_2, y_2)$$

$$d_3 R = d_3 k P = k d_3 P = k Q_3 = (x_3, y_3)$$

となる。また、上述した理由により、 $d_2 R$ 、 $d_3 R$ のx座標は、それぞれ x_2 、 x_3 となる。

【0411】したがって、図26に示すハッシュ値生成装置で生成されたハッシュ値 $h(x_2)$ 、 $h(x_3)$ と、図21および図25に示すハッシュ値生成装置で生成されたハッシュ値 $h(x_2)$ 、 $h(x_3)$ は各々一致するので、図26に示すしきい値逆ブロック部2612で求めたハッシュ値 $h(x_1)$ 、 $h(x_4)$ は、図21および図25に示すハッシュ値生成装置で生成されたハッシュ値 $h(x_1)$ 、 $h(x_4)$ と各々一致する。

【0412】復号処理部2613は、しきい値逆ブロック部2612で求めたハッシュ値 $h(x_1)$ を鍵2613として、暗号文2108aの4番目のデータ C_{12112} に対して復号・伸長処理を行う。これにより、平文2106の最初のNビットデータ2107を生成する。

【0413】さらに、復号処理部2617は、ハッシュ値生成装置2615において、鍵2613を基に生成されたハッシュ値を新しい鍵2616として、暗号文2108aの5番目のデータに対して復号・伸長処理を行う。これにより、平文2106の2番目のNビットデータを生成する。

【0414】上記の処理を、暗号文2108aを構成する最後のデータまで、順次繰り返すことで、平文2106を復元することができる。

【0415】なお、本発明の第五実施形態は、暗号文に f_1 、 f_2 、 \dots 、 f_n を含めることで、n人の相手に対してマルチキャストを行い、そのn人のうちの何人かは単独で復号可能とし、残りの者は少なくとも二人が協力することで、復号可能とすることもできる。あるいは、n人各々が単独で復号可能とすることもできる。

【0416】図24は、本発明の第五実施形態を構成するデータ暗号化装置のしきい値ロジック部の変形例の機能構成を示す図である。

【0417】図24に示すように、しきい値ロジック部2125aには、3つのデータ x_{12402} 、 x_{22403} 、 x_{32404} が入力される。

【0418】ここで、 x_{12402} 、 x_{22403} 、 x_{32406} は、それぞれ、図21に示す整数倍演算部2114、2115、2116で生成されたx座標値である。

【0419】演算部2408は、次式で示される処理を実行することで、データ f_{12406} を生成する。

$$【0420】f_1 = x_1 - h(x_2)$$

ここで、 $h(x_2)$ は、図21に示すハッシュ値生成装置2126で生成されたハッシュ値である。

【0421】また、演算部2410は、次式で示される処理を実行することで、データ f_{22407} を生成する。

$$【0422】f_2 = x_1 - h(x_3)$$

ここで、 $h(x_3)$ は、図21に示すハッシュ値生成装置2127で生成されたハッシュ値である。

【0423】図24に示すしきい値ロジック部2125aを用いることで、図21に示すデータ暗号化装置2101は、暗号化処理途中に生成される乱数 k と公開鍵 Q_2 2104とを基にデータ f_{12406} を生成することになる。このため、公開鍵 Q_2 2104と対の関係にある秘密鍵 d_2 の持ち主も、暗号文2108を単独で復号することができる。

【0424】同様に、暗号化処理途中に生成される乱数 k と公開鍵 Q_3 2105とを基に、データ f_{22407} を生成することになる。このため、公開鍵 Q_3 2105と対の関係にある秘密鍵 d_3 の持ち主も、暗号文2108を単独で復号することができる。

【0425】すなわち、暗号文2108を秘密鍵 d_1 、 d_2 、 d_3 のそれぞれの持ち主に同報通信（マルチキャスト）することで、これ等の持ち主各々が単独で復号することが可能となる。また、通信相手を一人増やすには、単に、ハッシュ値の長さ分（たとえば、80ビット）のデータを増やせばよいので、同報暗号通信を効率的に行うことができる。ここで、ハッシュ関数には一方向性だけが要求され、衝突回避性は要求されない。

【0426】なお、上記の実施形態では、楕円曲線暗号として、

$$y^2 = x^3 + ax + b$$

に基づくものを用いたが、その代わりに、楕円曲線暗号として、 $y^2 + xy = x^3 + ax + b$ に基づくものを用いてもよい。

【0427】

【発明の効果】以上説明したように本発明によれば、データ攪乱度の高いハッシュ値を、迅速に生成することができる。

【0428】また、データ攪乱度の高い鍵や暗号文などのデータを、迅速に生成することができる。

【0429】さらに、同じデータを複数の宛先に暗号化して送信しようとする場合に、各宛先毎に、当該宛先から予め配布されて公開鍵を用いて暗号化処理を行わなければならないなくなる。

【0430】また、間違ってファイルから復号鍵を消去するなどして、復号鍵を紛失してしまった場合でも、自己宛てに送られてきた暗号化データを、他の二人以上の受信者が協力することで復号することが可能になる。

【図面の簡単な説明】

【図1】本発明の第一実施形態であるハッシュ値生成装

置の機能構成を示す図である。

【図 2】図 1 に示す混合処理部 103 での処理の一例を説明するための図である。

【図 3】図 1 に示す伸長処理部 104 での処理の一例を説明するためのフロー図である。

【図 4】図 1 に示す単射拡大部 105 での処理の一例を説明するためのフロー図である。

【図 5】図 4 に示すステップ 405 における 64 ビットデータから 96 ビットデータへの単射拡大処理の一例を説明するためのフロー図である。

【図 6】図 4 に示すステップ 406 における 96 ビットデータから 128 ビットデータへの単射拡大処理の一例を説明するためのフロー図である。

【図 7】図 4 に示すステップ 407 における 128 ビットデータから 256 ビットデータへの単射拡大処理の一例を説明するためのフロー図である。

【図 8】本発明の第一実施形態の変形例であるハッシュ値生成装置の機能構成を示す図である。

【図 9】図 8 に示す混合処理部 801 での処理の一例を説明するための図である。

【図 10】図 8 に示す単射拡大部 803 での処理の一例を説明するためのフロー図である。

【図 11】図 10 に示すステップ 1005 における 64 ビットデータから 64 ビットデータへの単射処理の一例を説明するためのフロー図である。

【図 12】図 10 に示すステップ 1006 における 64 ビットデータから 80 ビットデータへの単射拡大処理の一例を説明するためのフロー図である。

【図 13】本発明の第二実施形態であるデータ暗号化装置の機能構成を示す図である。

【図 14】図 13 に示す π 関数処理部 1306 ~ 1313 における処理の一例を説明するためのフロー図である。

【図 15】本発明の第三実施形態であるマスキング装置の機能構成を示した図である。

【図 16】本発明の第四実施形態であるデータ暗号化システムを構成するデータ暗号化装置の機能構成を示す図である。

【図 17】図 16 に示す圧縮・暗号化部 1612、1615、・・・の機能構成を示す図である。

【図 18】本発明の第四実施形態であるデータ暗号化システムを構成するデータ復号化装置の機能構成を示す図である。

【図 19】図 18 に示す復号・伸長部 1810、1812、・・・の機能構成を示す図である。

【図 20】図 16 に示す本発明の第四実施形態におけるデータ暗号化装置の変形例の機能構成を示す図である。

【図 21】本発明の第五実施形態であるデータ暗号化システムを構成するデータ暗号化装置の機能構成を示す図である。

【図 22】図 21 に示すしきい値ロジック部 2125 の機能構成を示す図である。

【図 23】図 21 に示す圧縮・暗号化部 2120、・・・の機能構成を示す図である。

【図 24】本発明の第五実施形態を構成するデータ暗号化装置のしきい値ロジックの変形例の機能構成を示す図である。

【図 25】本発明の第五実施形態を構成するデータ暗号化装置の変形例の機能構成を示す図である。

10 【図 26】本発明の第五実施形態であるデータ暗号化システムを構成するデータ復号化装置の機能構成を示す図である。

【図 27】従来のハッシュ関数を説明するための図である。

【符号の説明】

101、101a、1301、1509、1809、1912、2011、2014、2119、2121、2126 ~ 2128、2323、2620、2623、2615 ハッシュ値生成装置

20 102 データ拡大部

103、801 混合処理部

104 伸長処理部

105、803 単射拡大部

1311、1601、1801、2001 データ暗号化装置

1306 ~ 1313、1513、1514、1518、1519、1710、1711、1716、1718、2315、2316、2322、2324 π 関数処理部

30 1501 マスキング生成装置

1508、1607、2007、2113 乱数生成装置

1608、1609、1807、2008、2009、2114 ~ 2116、2123、2124、2501、2619、2622 整数倍演算部

1610、1613、1808、1811 番号生成部

1612、1615、2012、2015、2120

圧縮・暗号化部

1707、1713、2312、2319 圧縮処理部

40 1706、1712、1717、1906、1909、

1915、2310、2317 拡張部

1810、1813 復号伸長部

1911、1917、2614、2617 伸長処理部

1907、1910、1914、1916 π^{-1} 関数処理部

2125 しきい値ロジック部

2325、2611 演算部

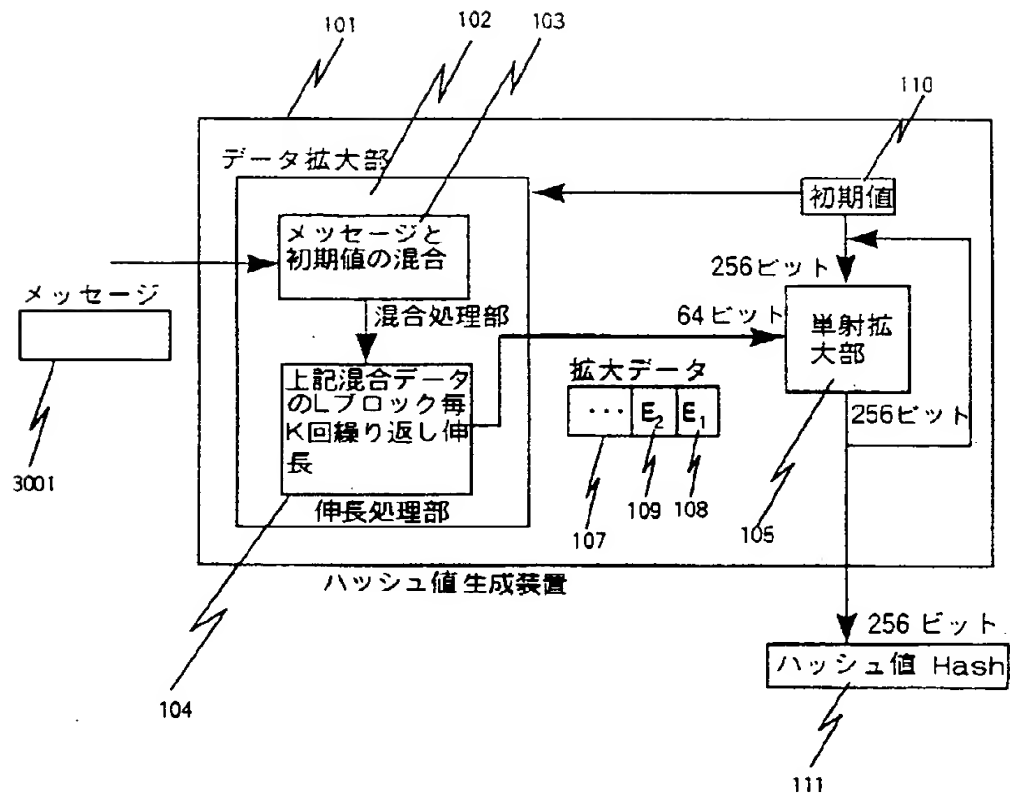
2607 データ回復機関

2608、2609 鍵保管機関

50 2612 しきい値逆ロジック部

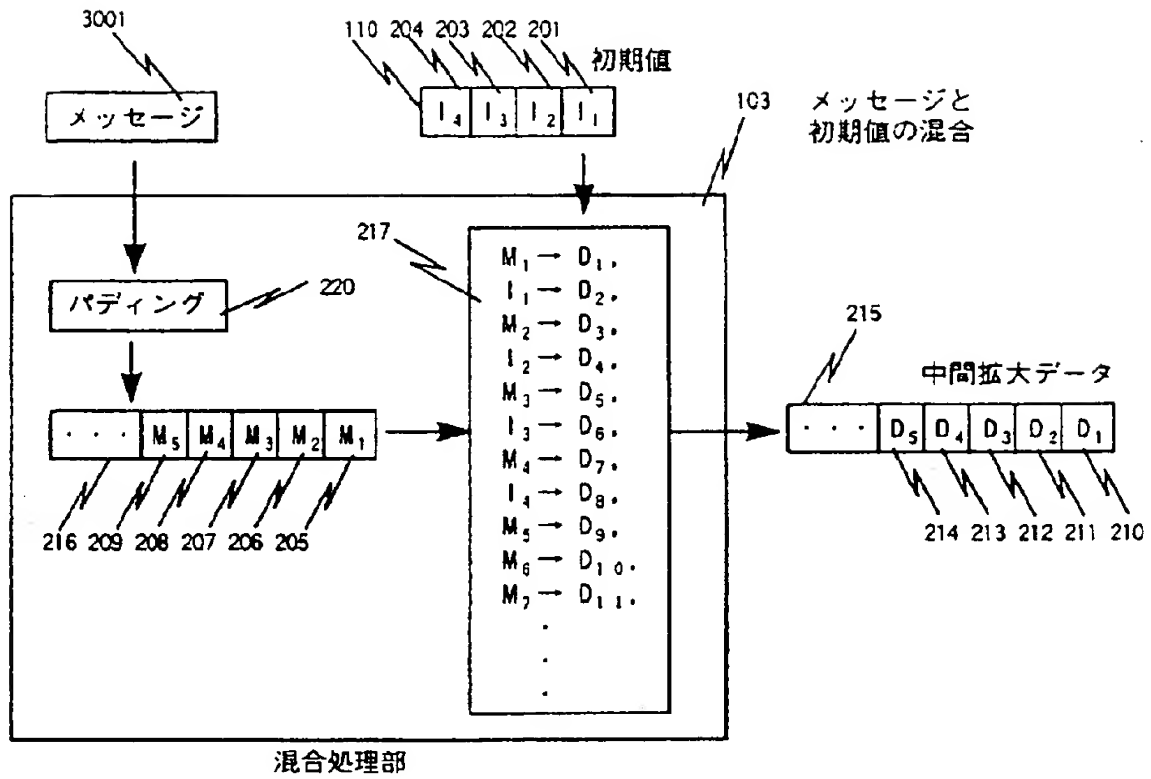
【図1】

図 1



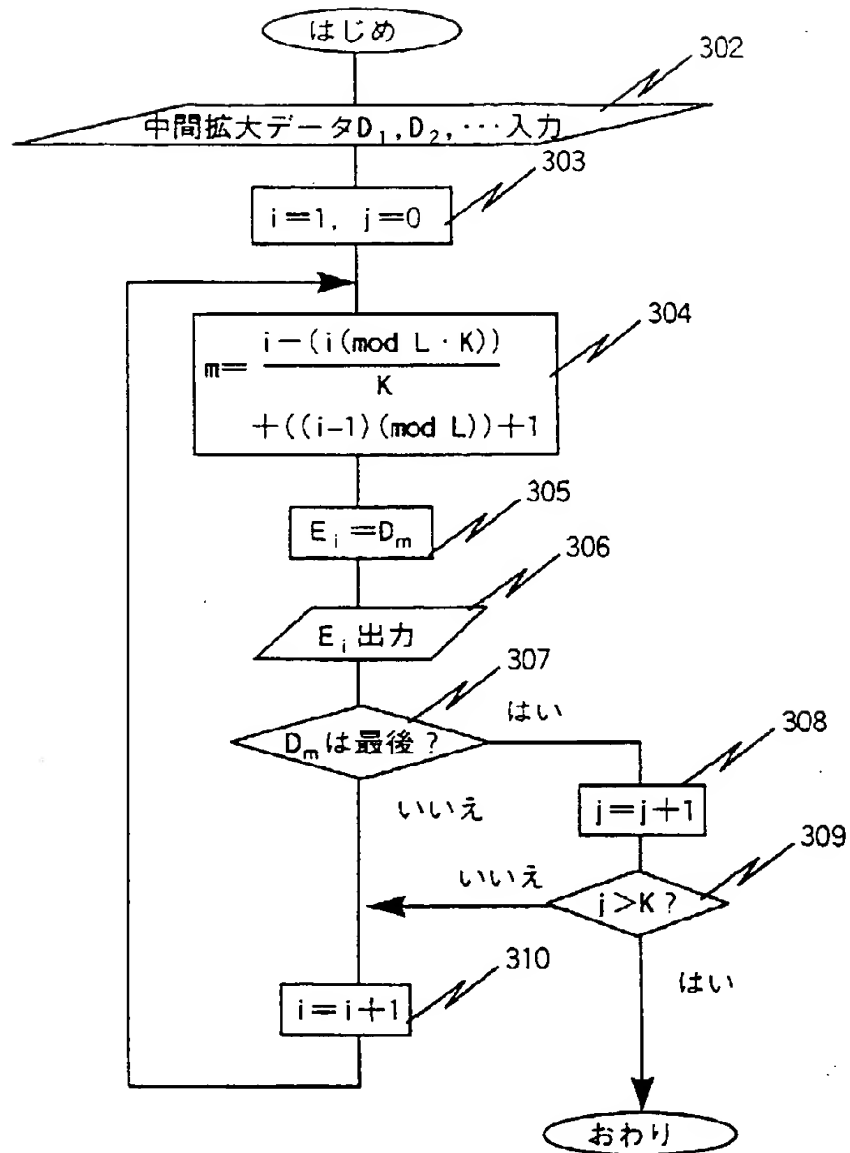
【図 2】

図2



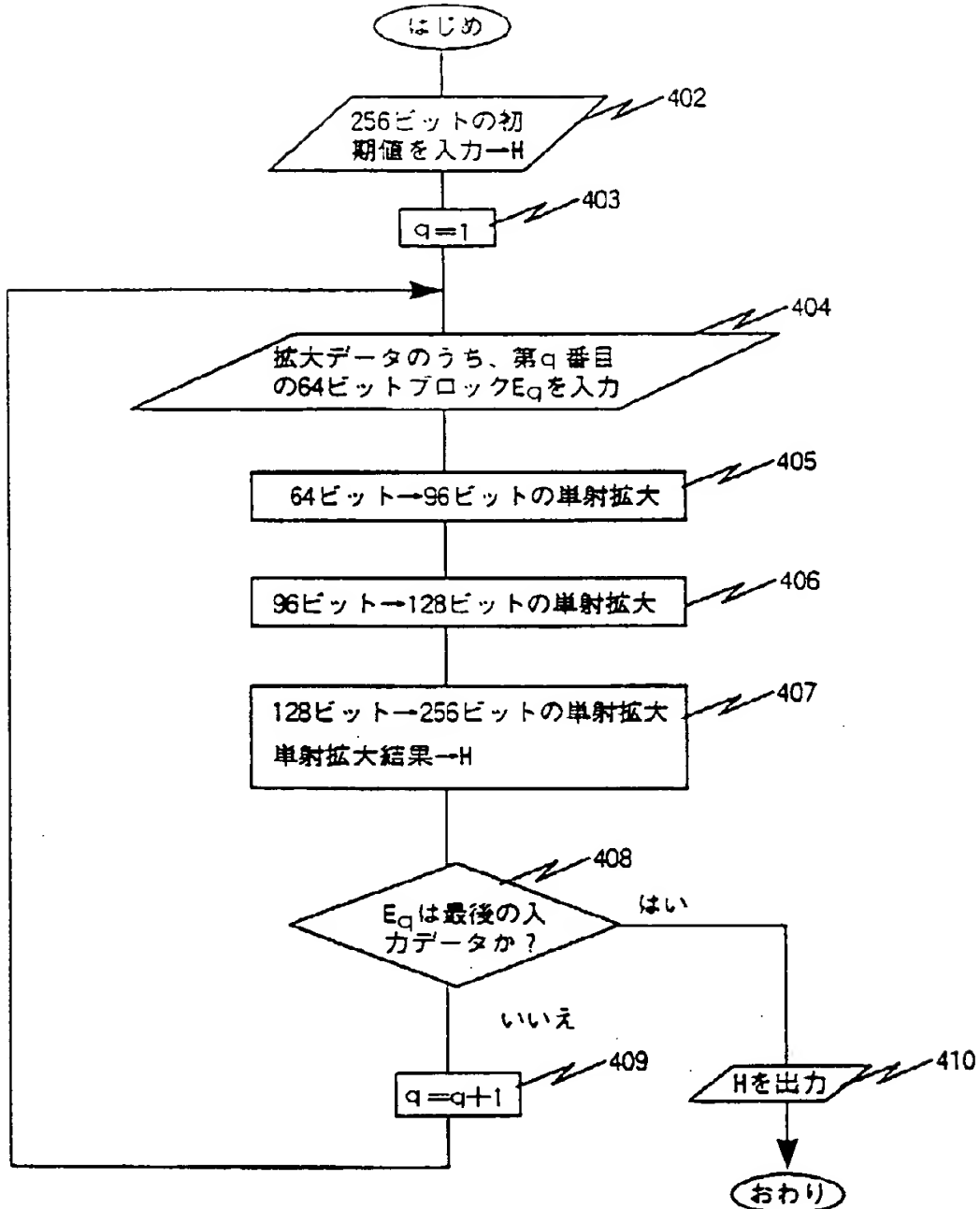
【図 3】

図 3



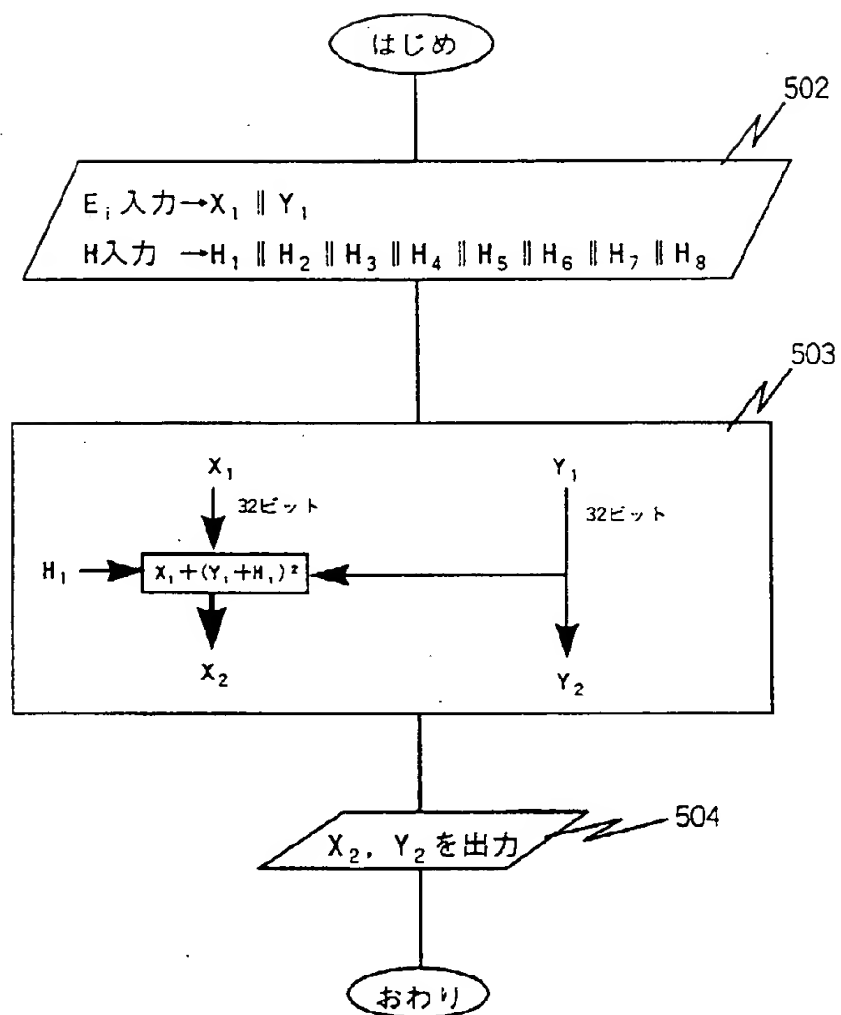
【図4】

図4

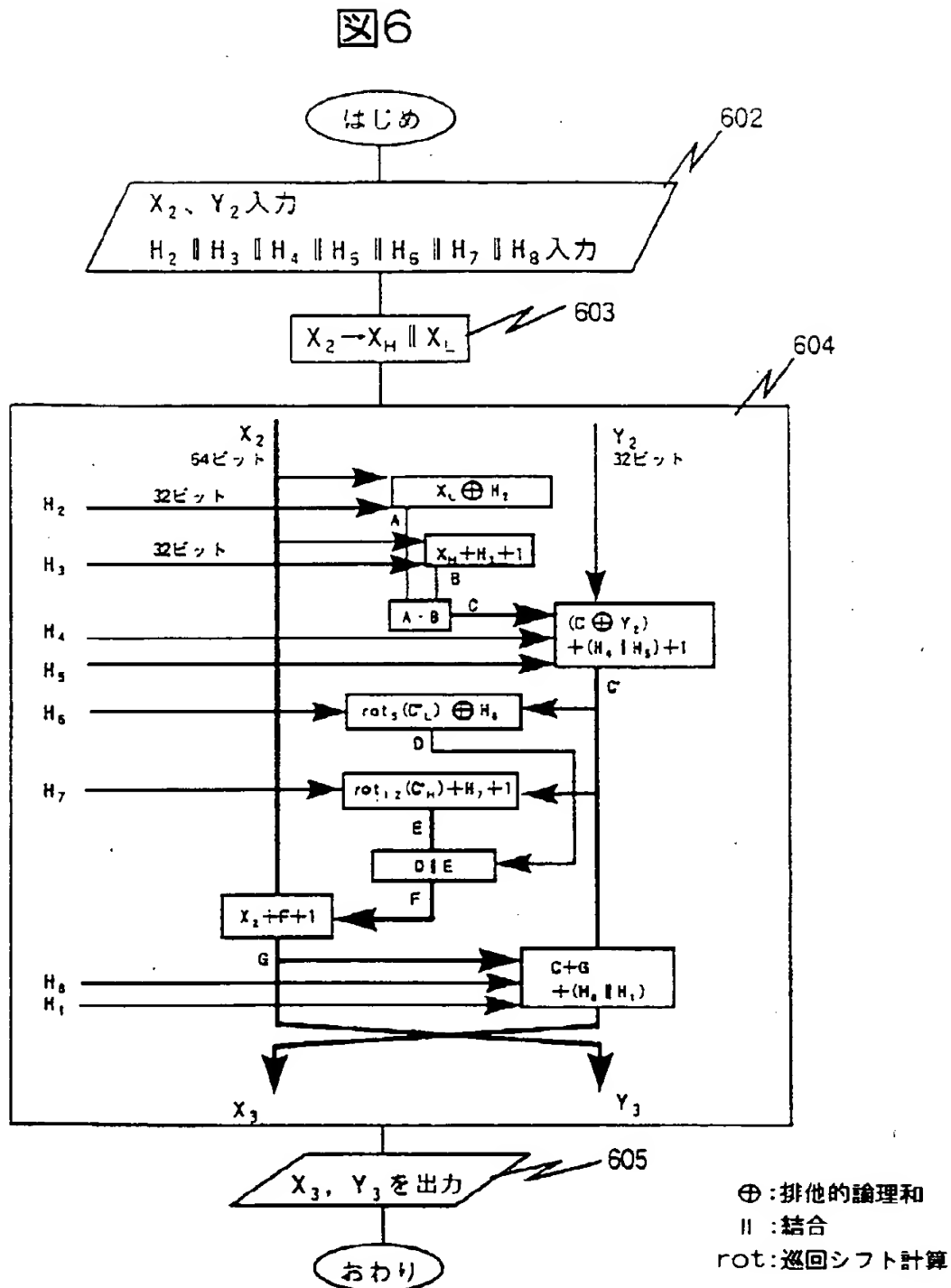


【図5】

図 5

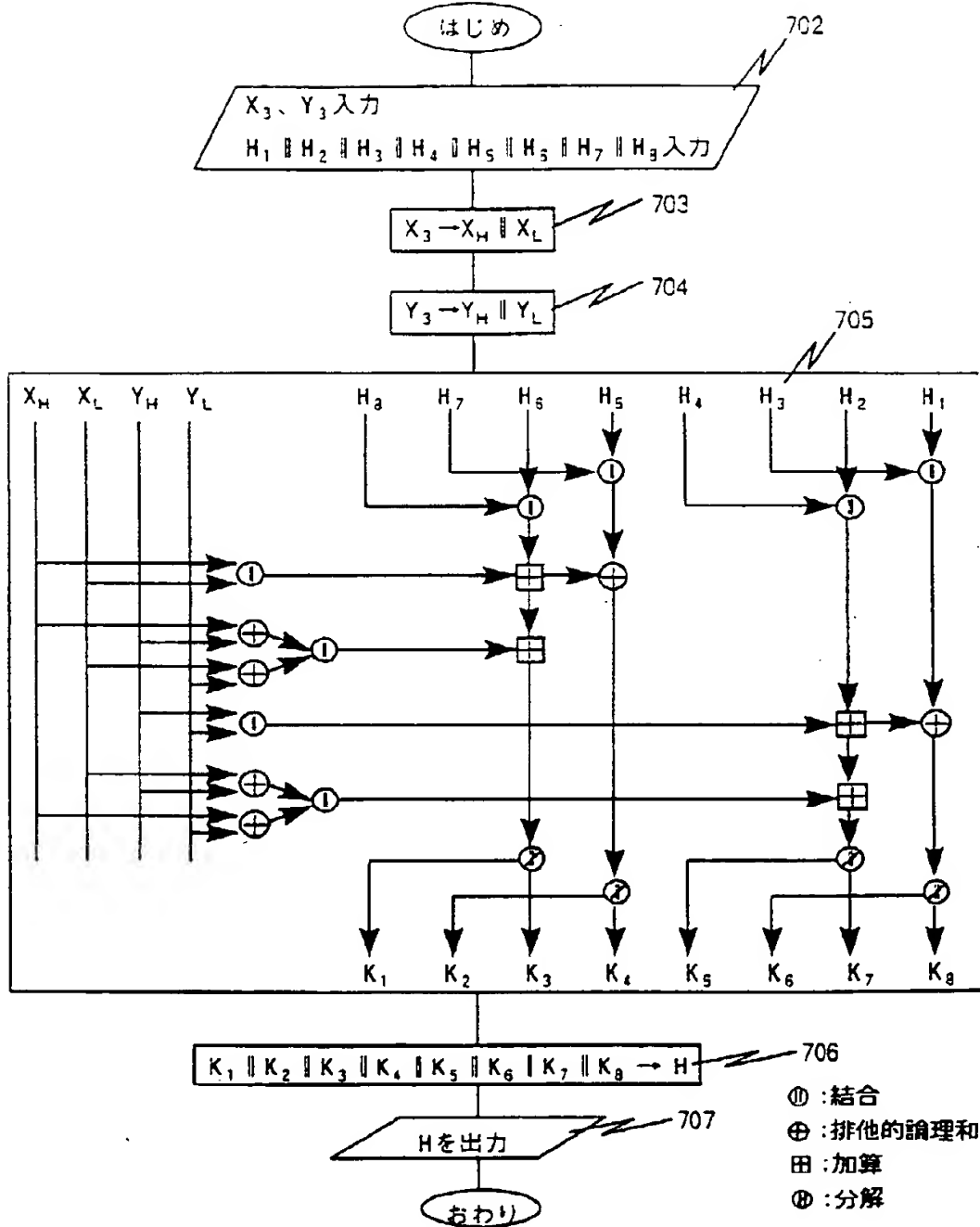


【図6】



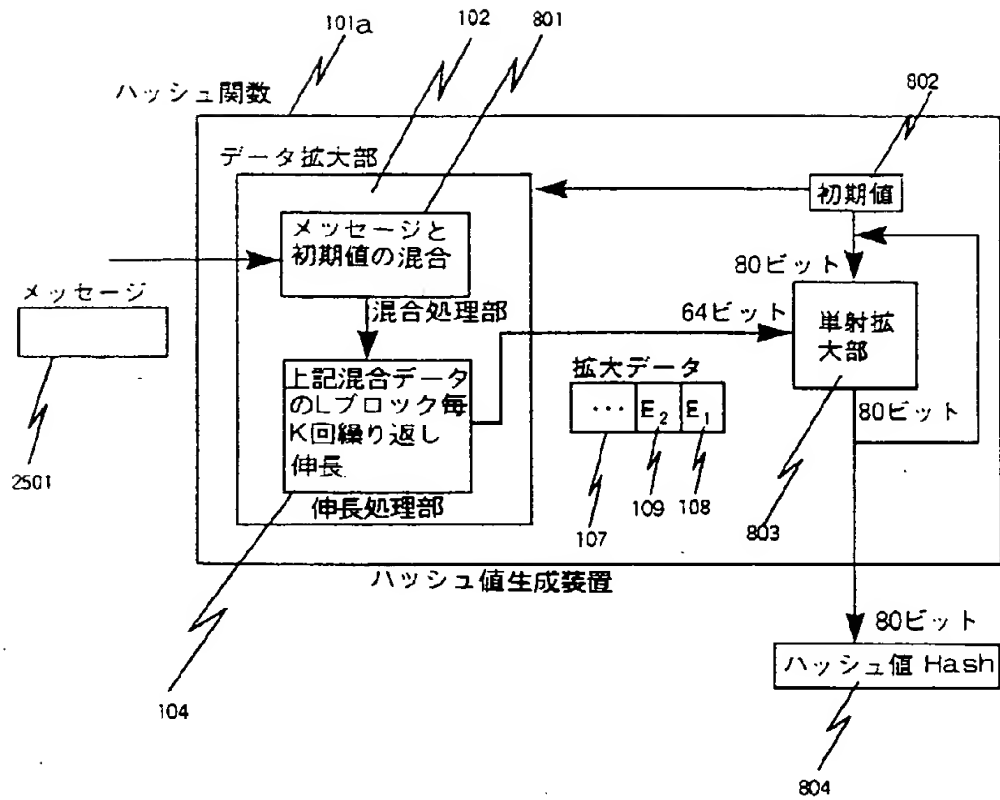
【図 7】

図 7



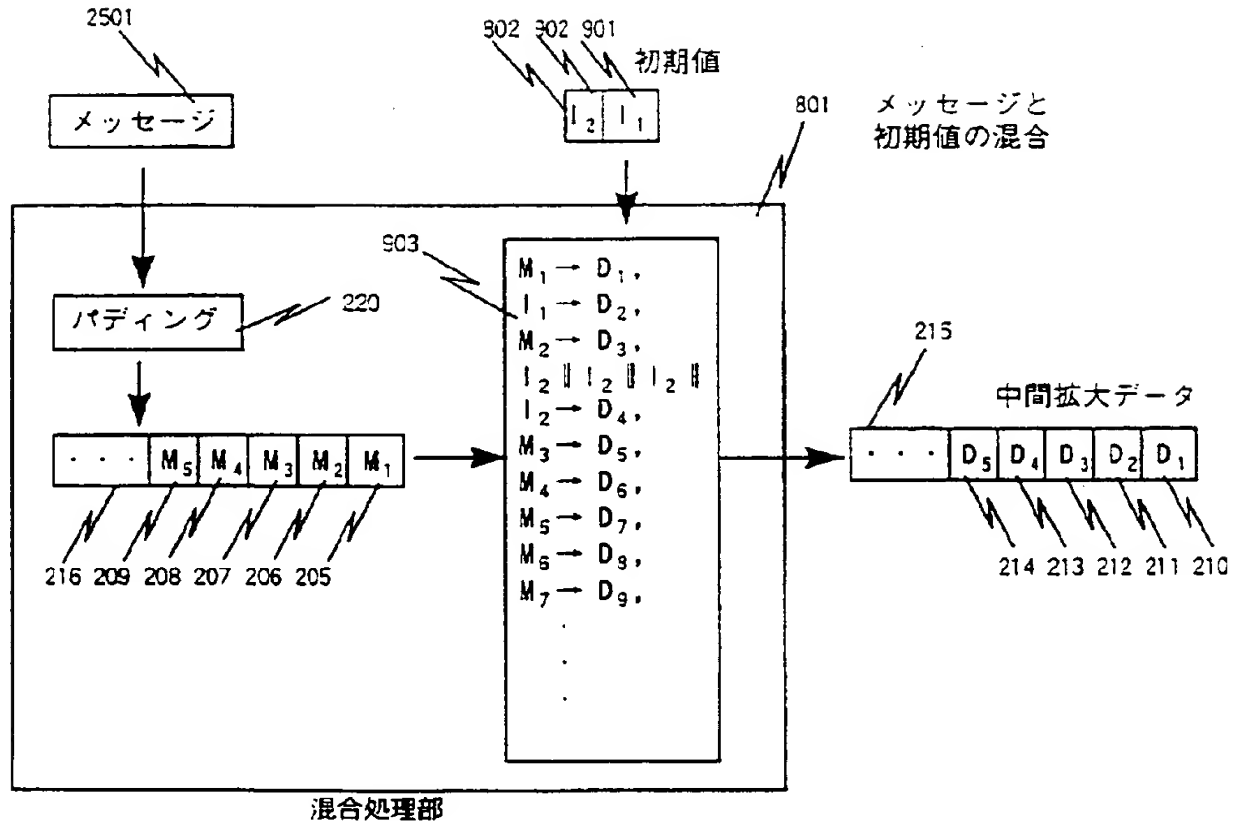
【図8】

図 8



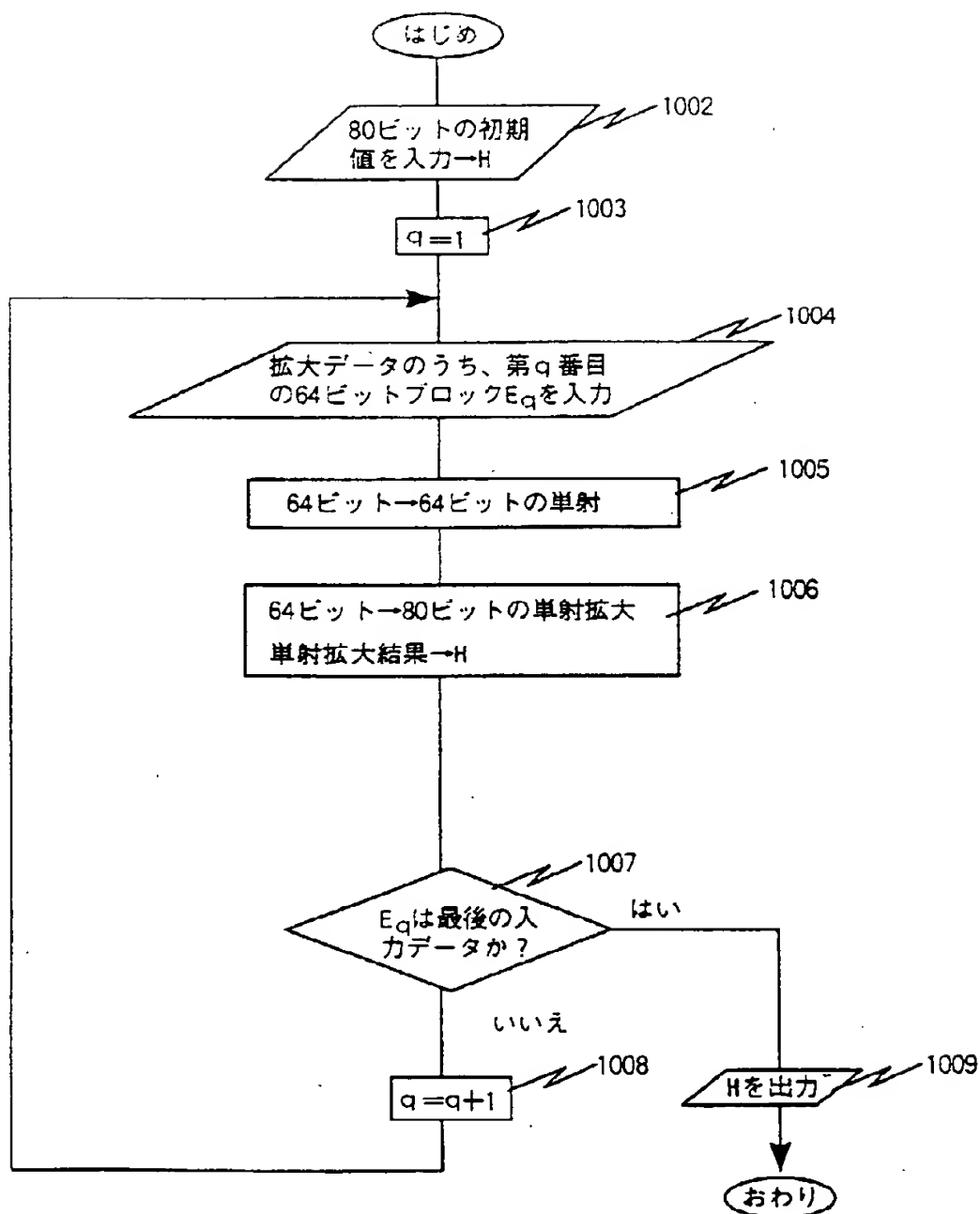
【図 9】

図 9



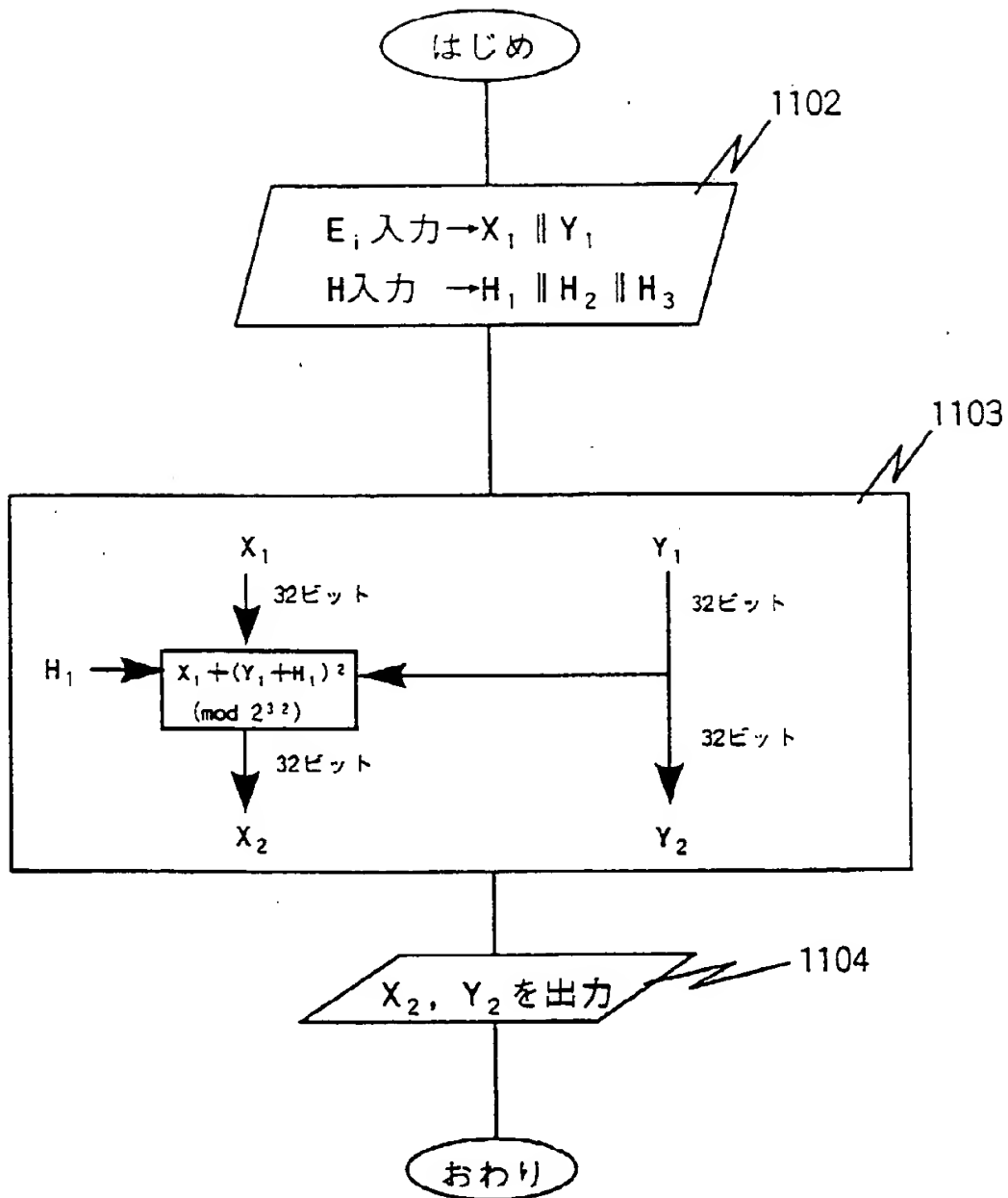
【図10】

図10

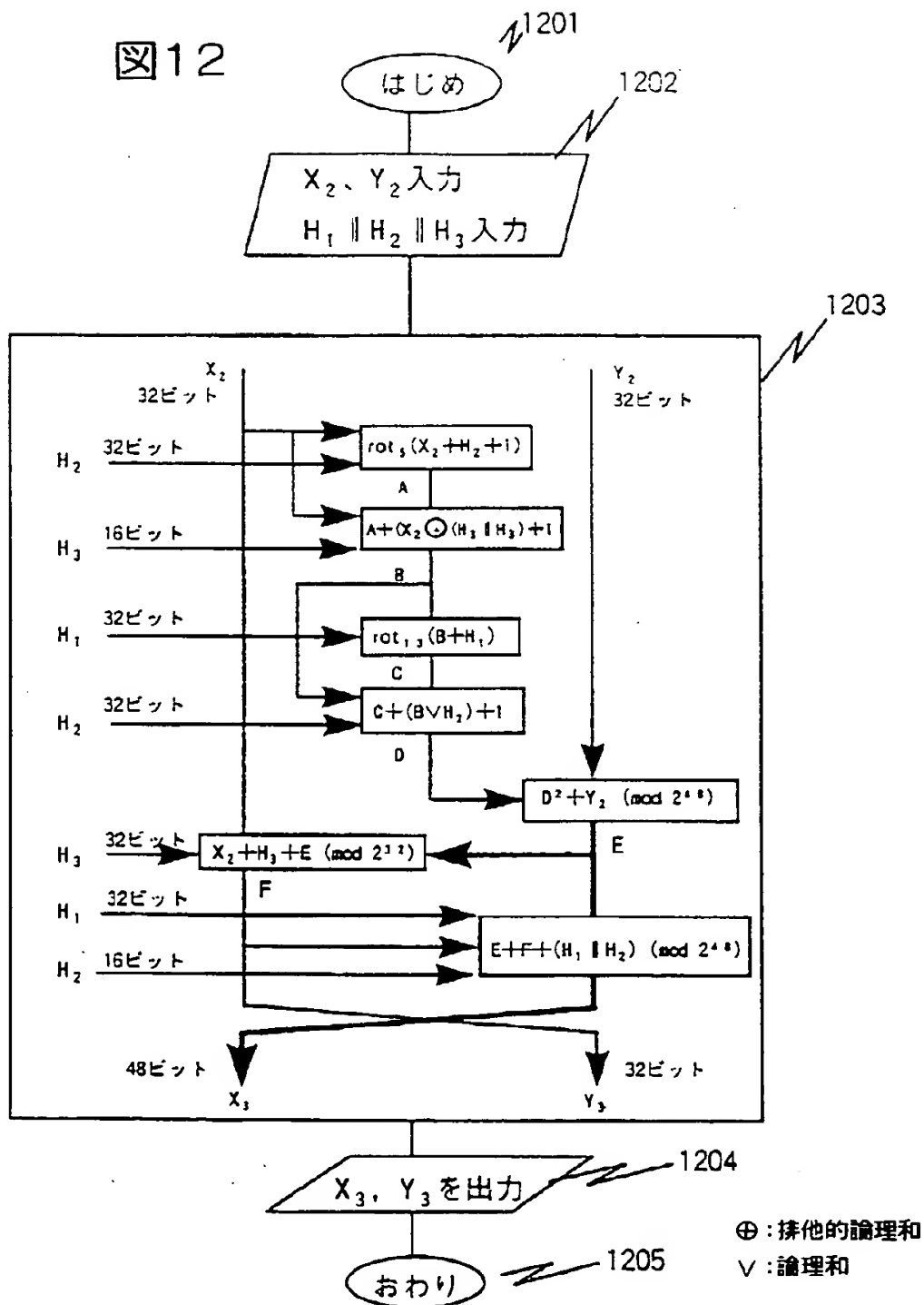


【図11】

図11

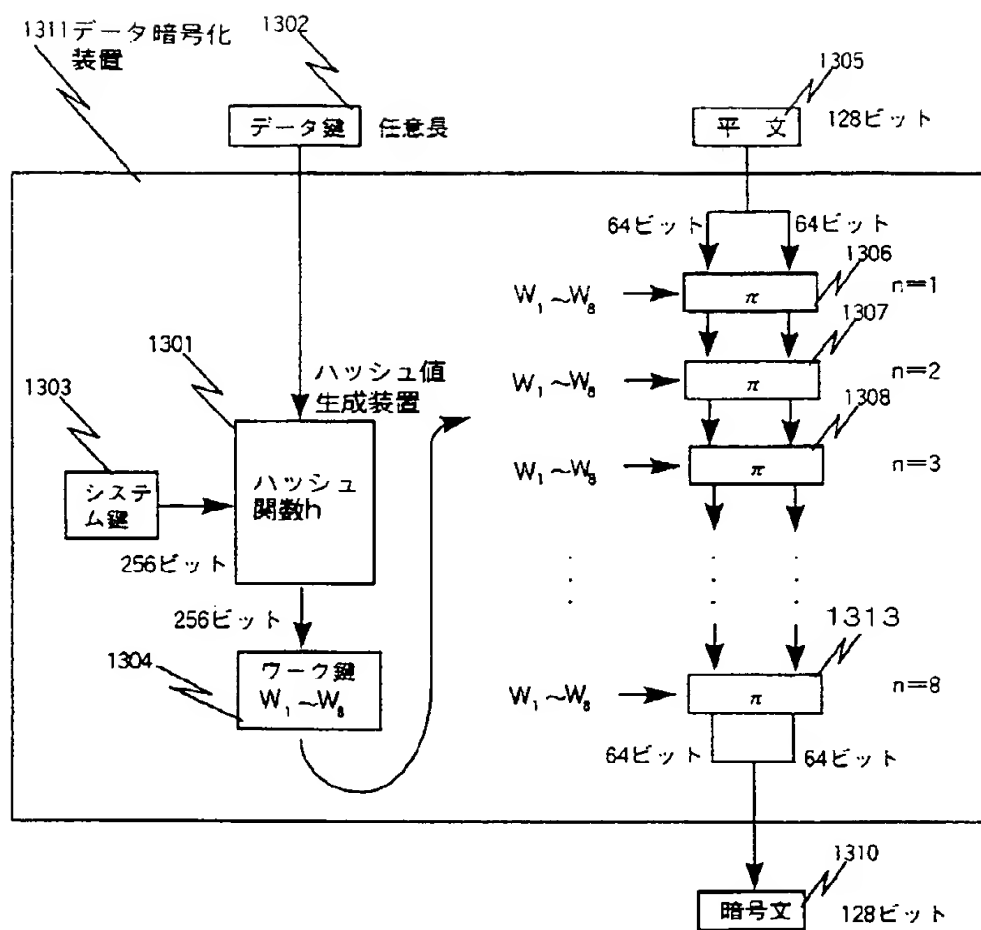


【図12】



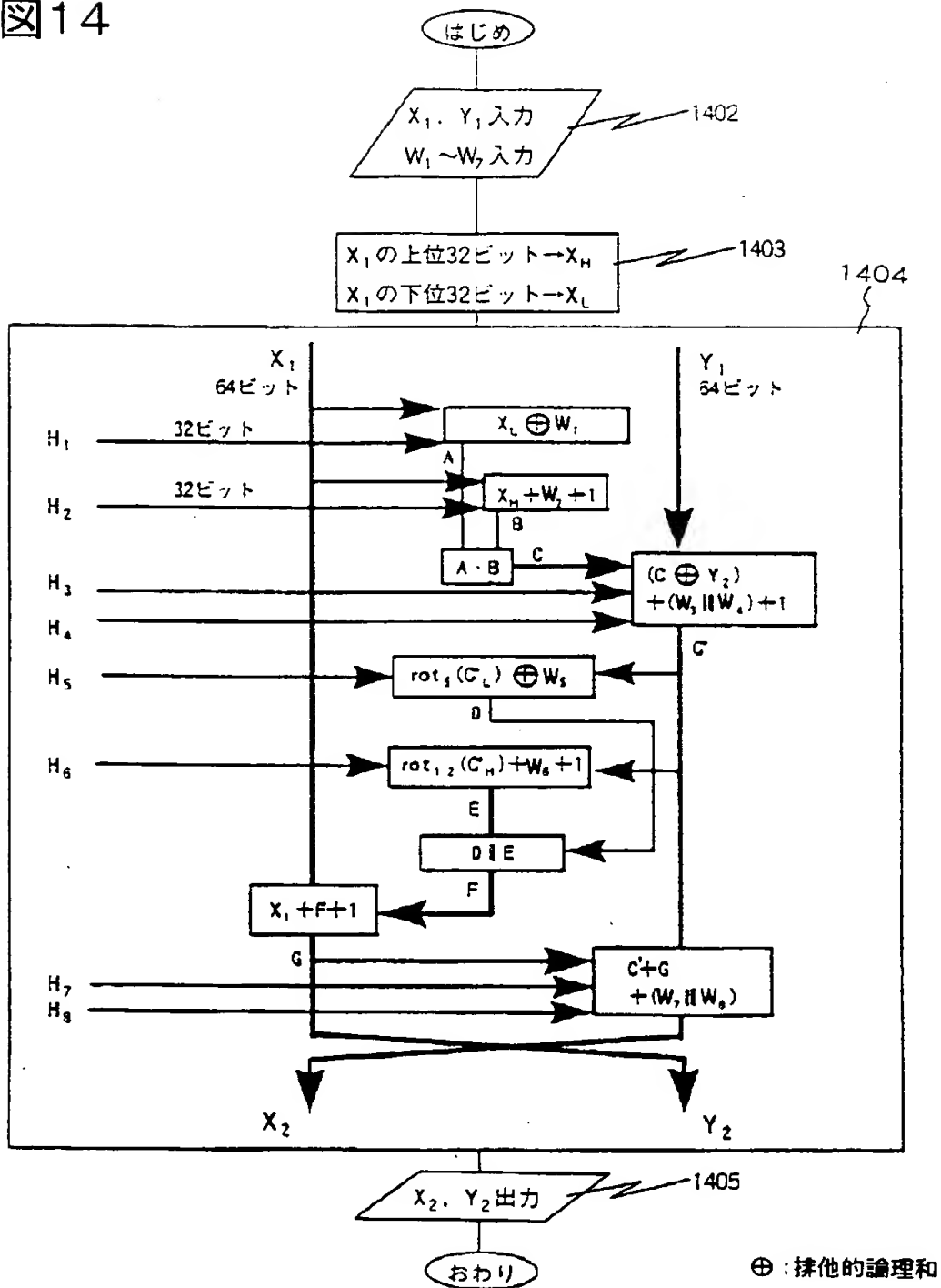
【図13】

図13



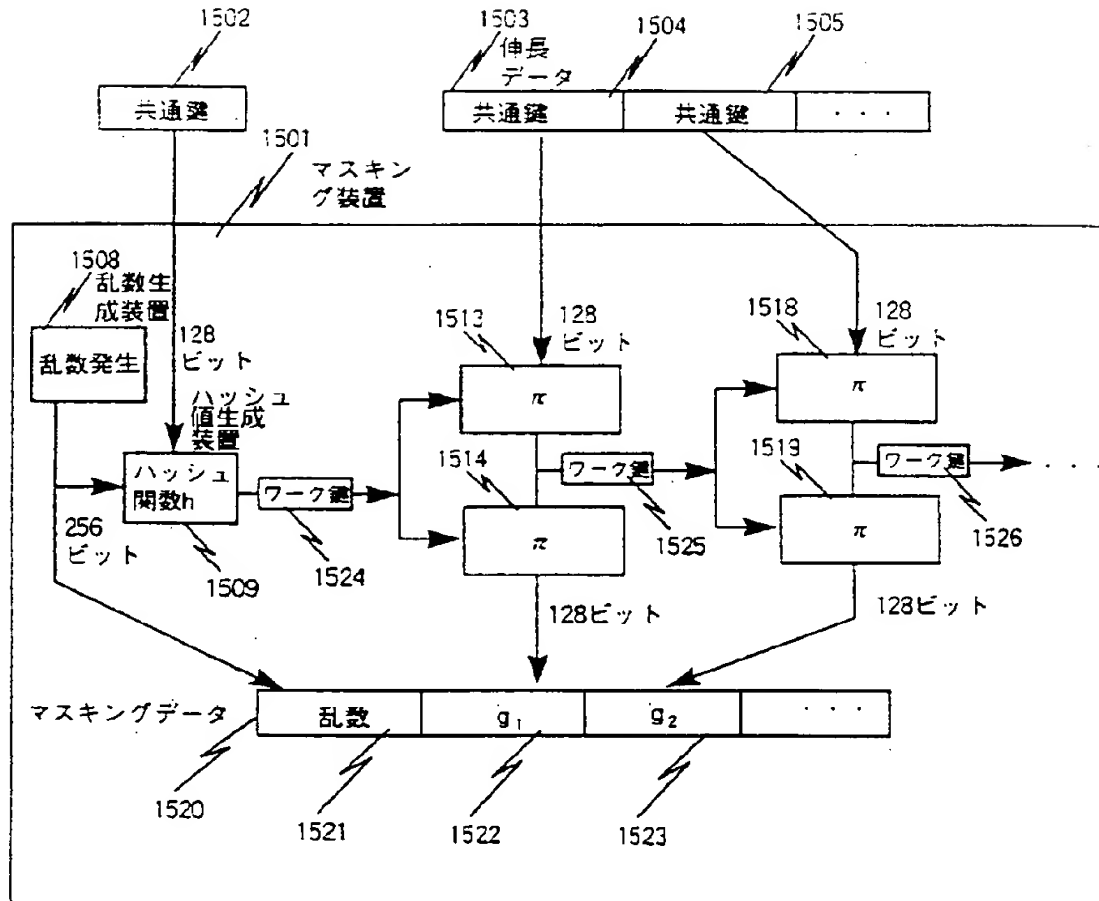
【図14】

図14



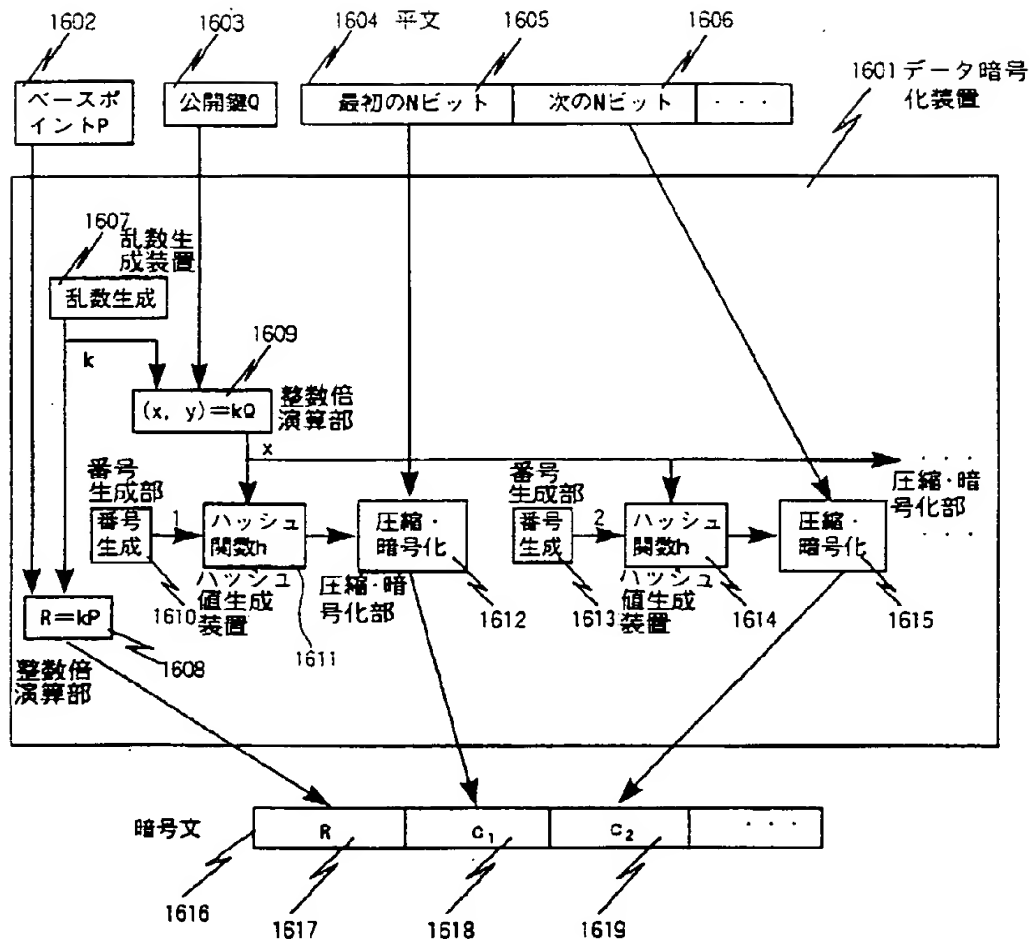
【図15】

図15



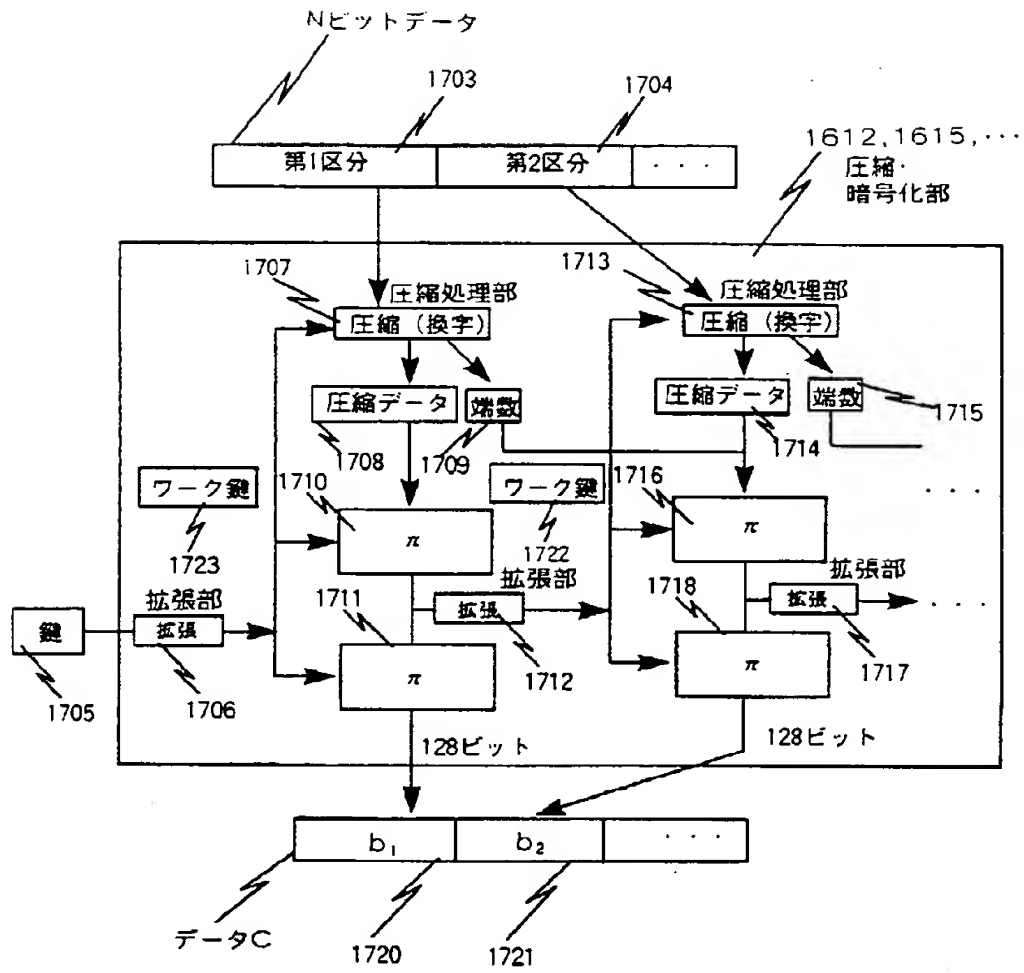
【図16】

図 16



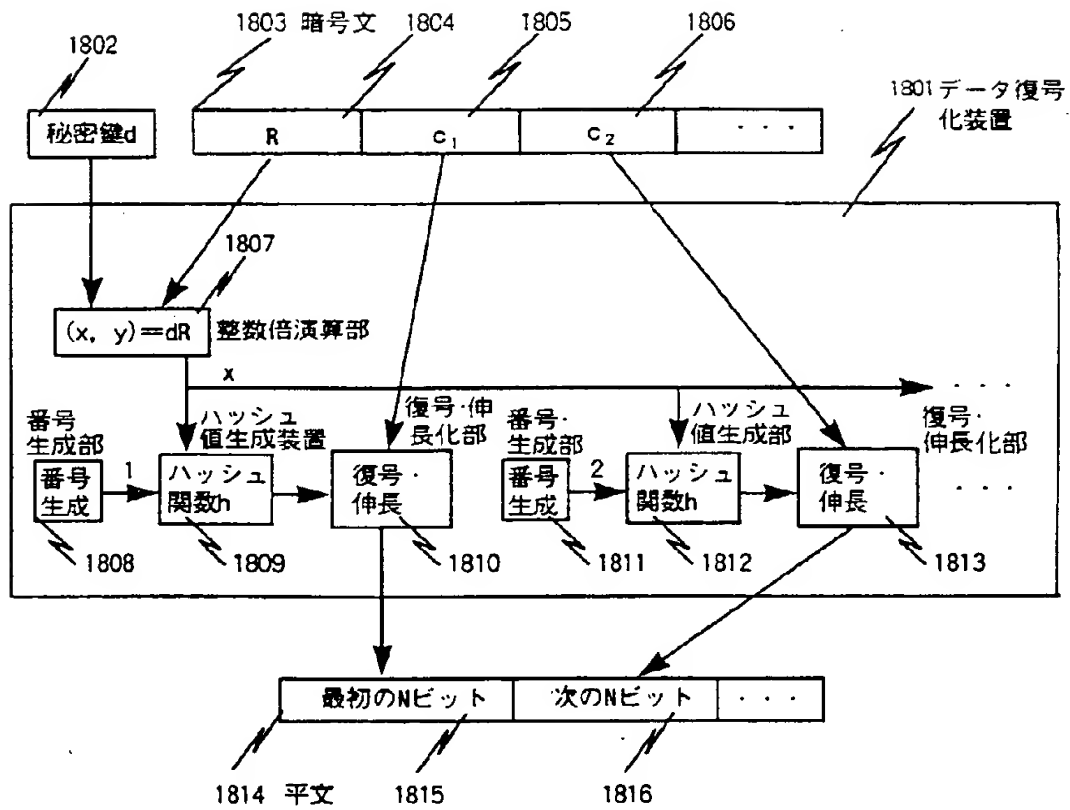
【図17】

図17



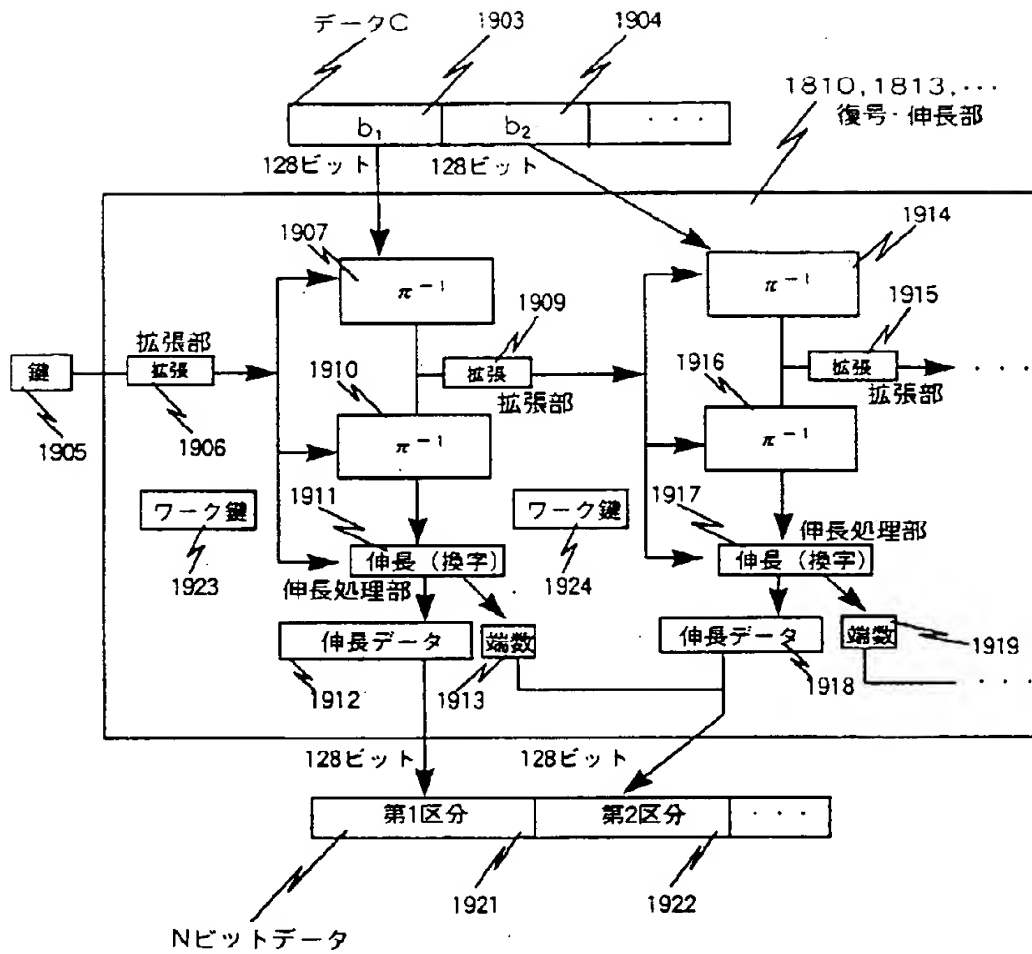
【図18】

図 18



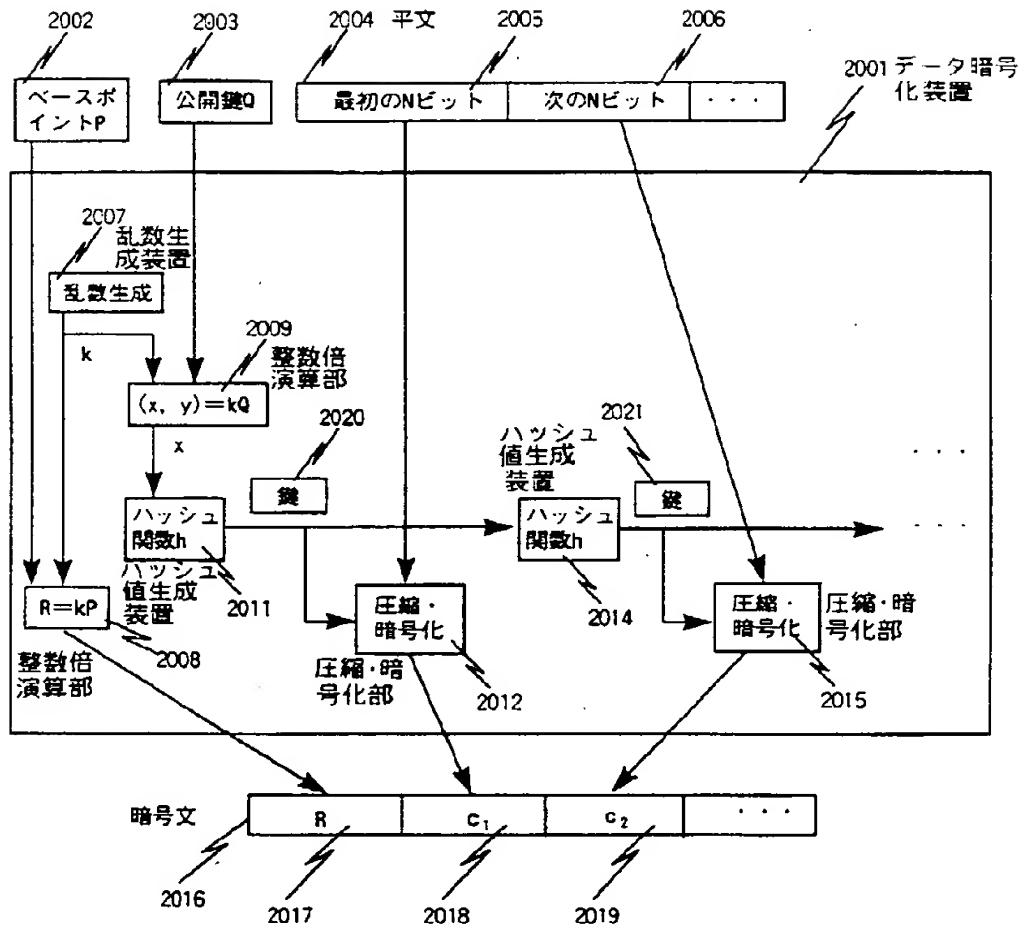
【図19】

図 19



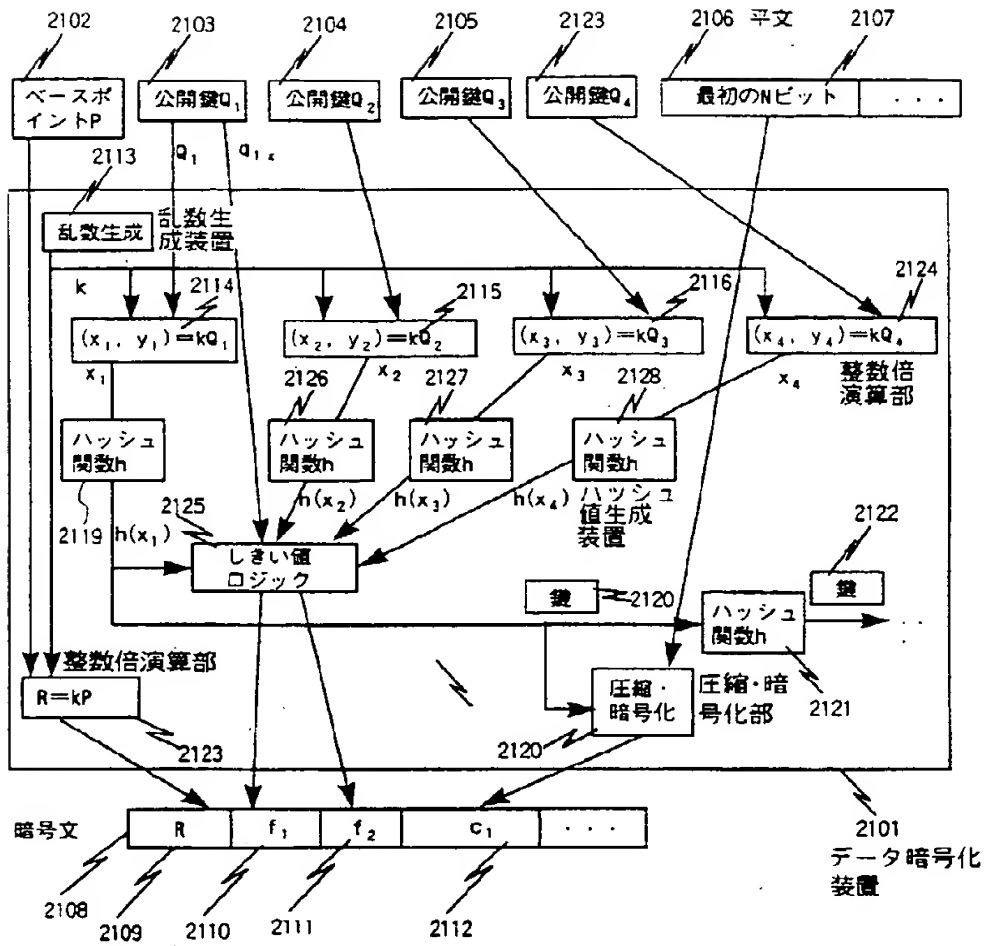
【図20】

図20

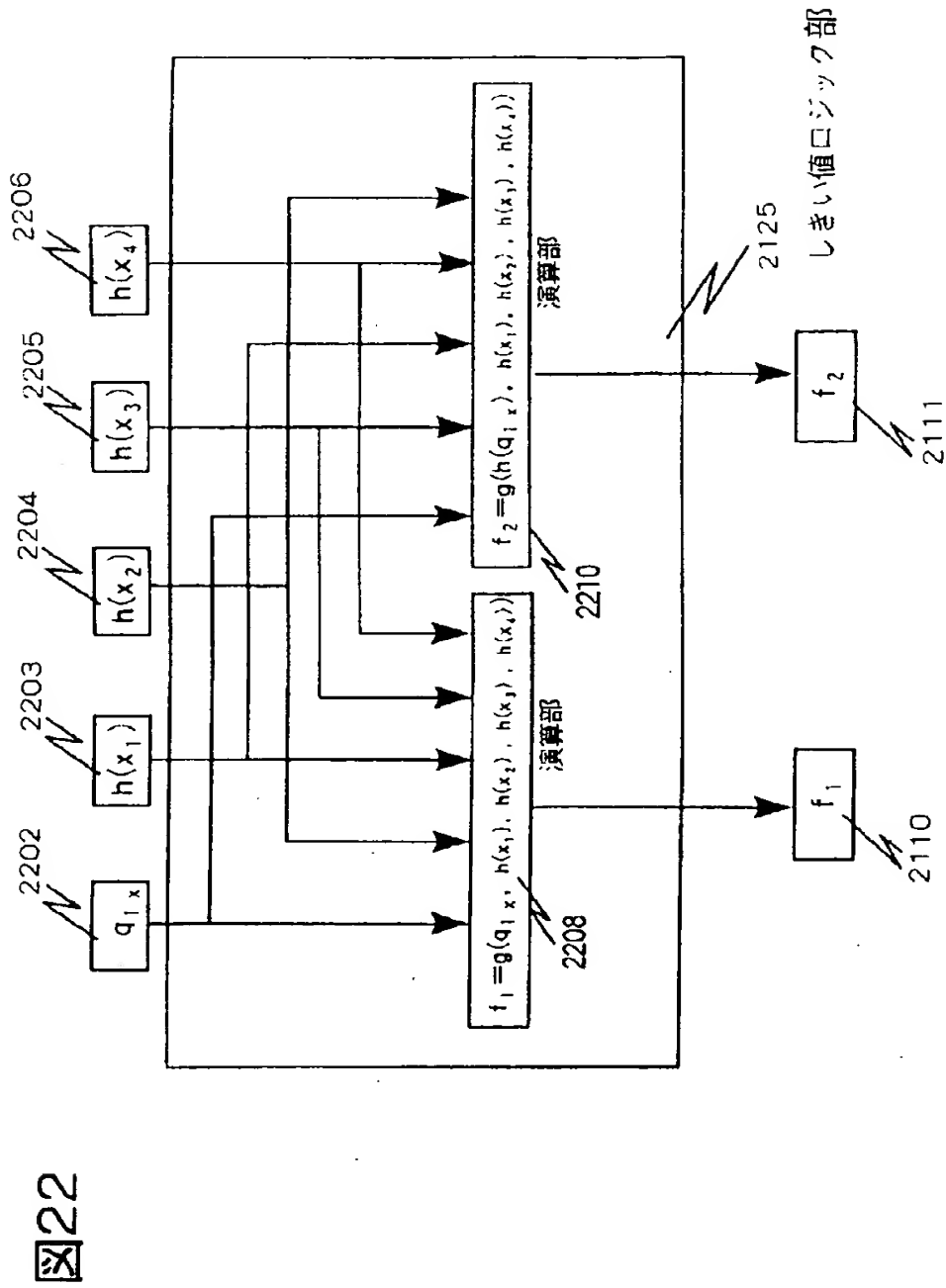


【図21】

図 2 1



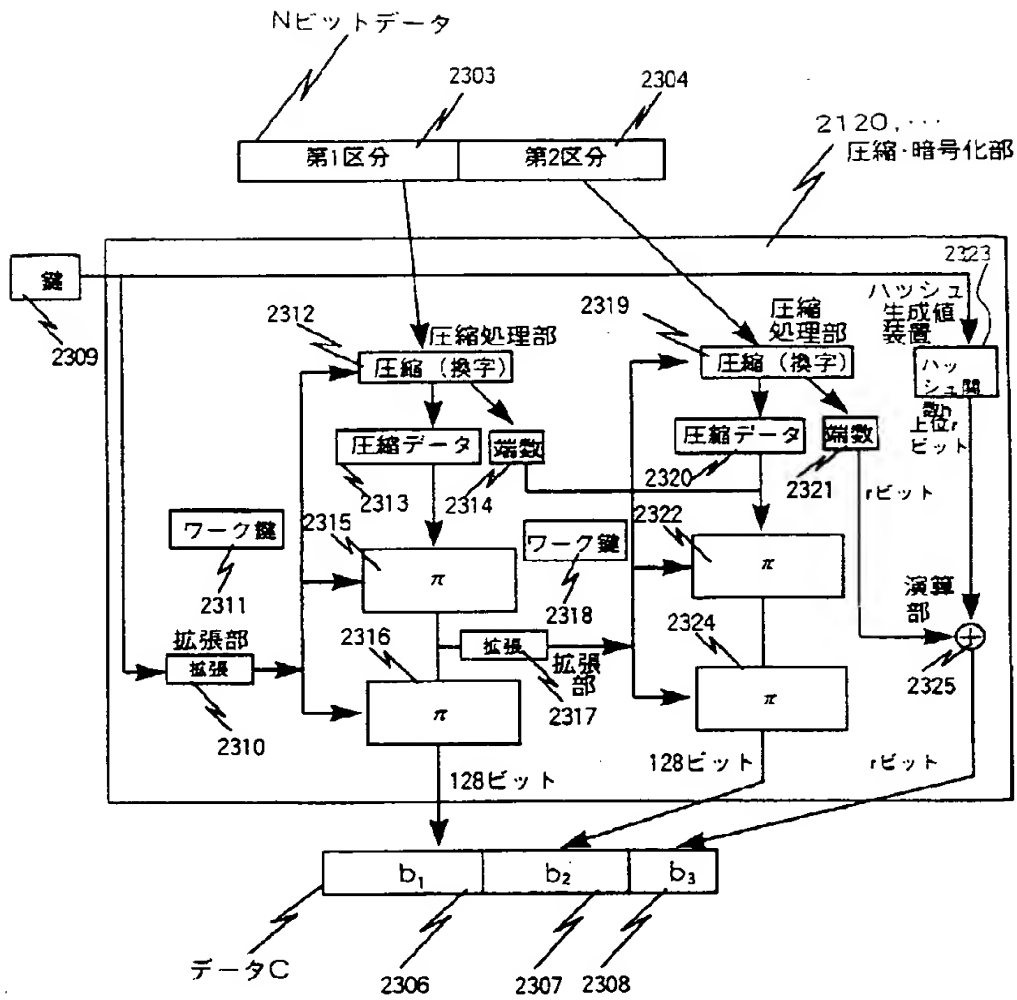
【図22】



$$g(x, a_1, a_2, a_3, a_4) \equiv a_1 + h(a_2) \cdot x + h(a_3) \cdot x^2 + h(a_4) \cdot x^3 \pmod{n}$$

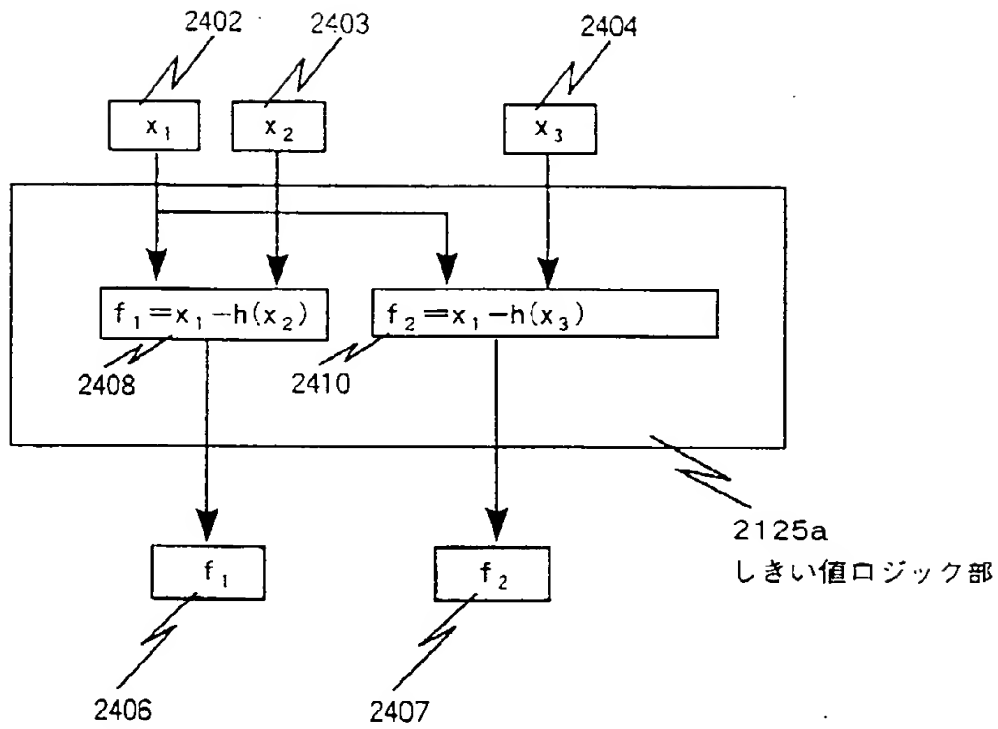
【図23】

図 2 3



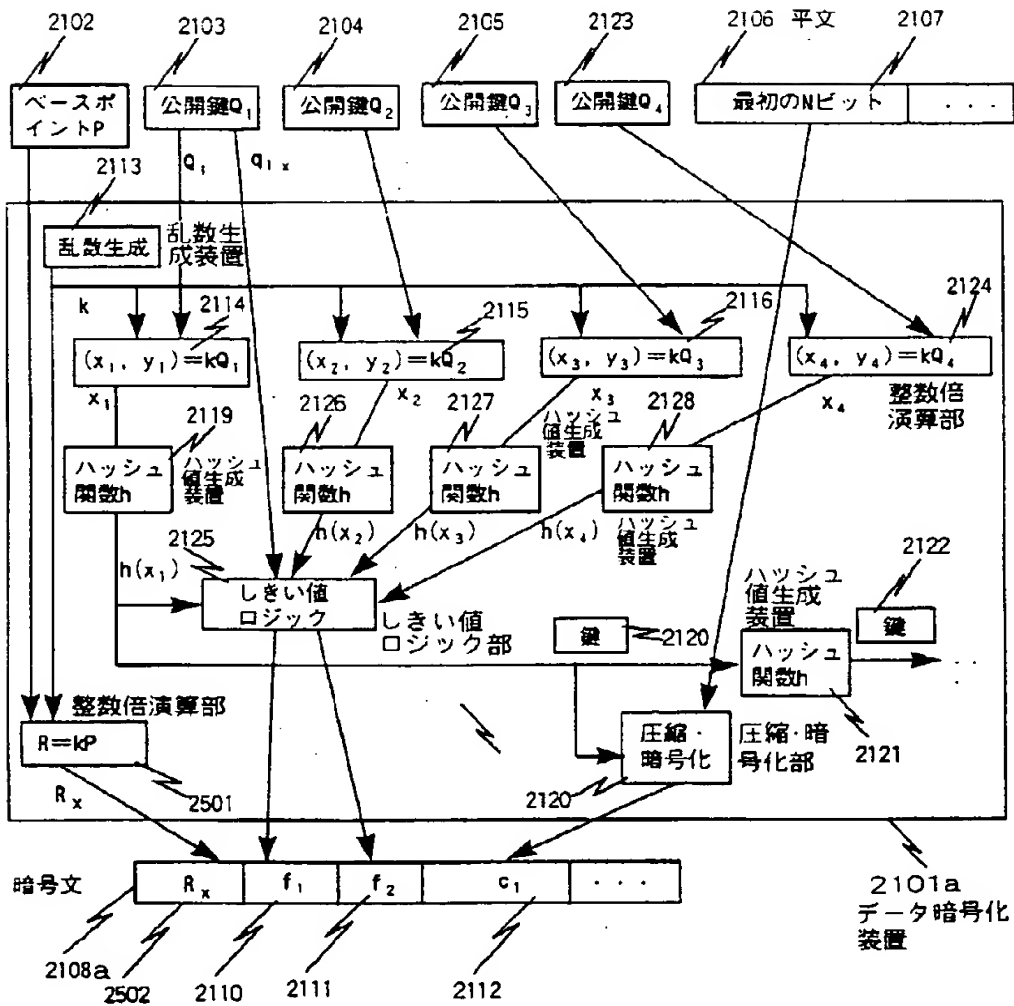
【図24】

図24



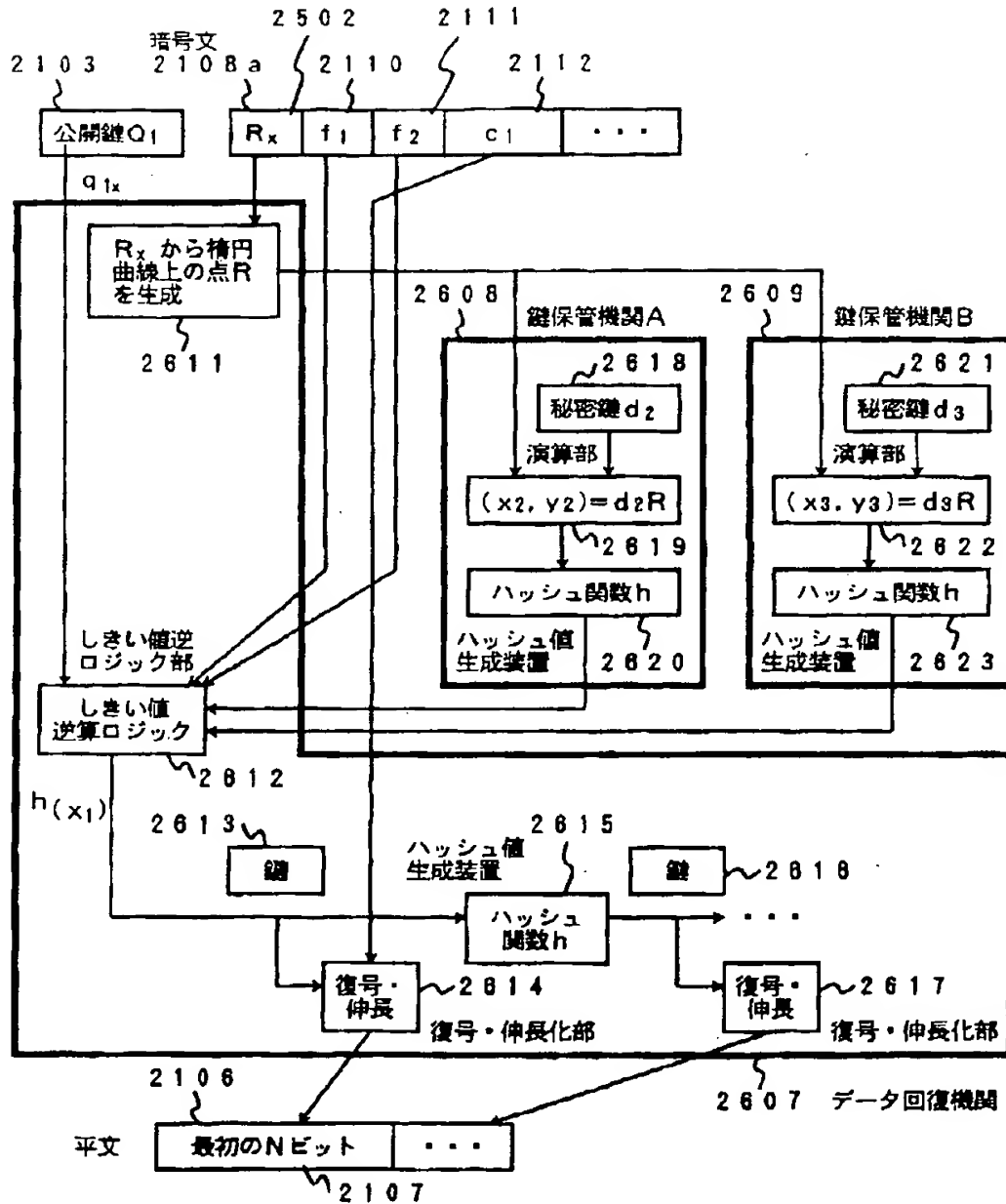
【図25】

図 25



【図26】

図26



【図27】

図27

